eport No. FHWA-RD-73-43

PREDICTION OF LONG-TERM STRESS RANGES
lser's Manual—Bridge Load Generator (BRIGLDI)

J.W. Fothergill, H.Y. Lee, and P.A. Fothergill

June 1973
Final Report

37

Prepared for
FEDERAL HIGHWAY ADMINISTRATION
Offices of Research & Development
Washington, D.C. 20590

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The contents of this report reflect the views of the contracting organization, which is responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policy of the Department of Transportation. This report does not constitute a standard, specification, or regulation.

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA-RD-73-43		

4. Title and Subtitle	5. Report Date
PREDICTION OF LONG-TERM STRESS RANGES User's Manual - Bridge Load Generator	June 1973 6. Performing Organization Code
7. Author(s)	**8. Performing Organization Report No.**
J. W. Fothergill, H. Y. Lee, and P. A. Fothergill	
9. Performing Organization Name and Address	**10. Work Unit No.**
Integrated Systems, Inc. 6900 Wisconsin Avenue Chevy Chase, Maryland 20015	FCP 35G1-012 **11. Contract or Grant No.** DOT-FH-11-7904
12. Sponsoring Agency Name and Address	**13. Type of Report and Period Covered**
Offices of Research and Development Federal Highway Administration U. S. Department of Transportation Washington, D. C. 20590	Final Report **14. Sponsoring Agency Code**

15. Supplementary Notes

FHWA contract manager: W. L. Armstrong (HRS-11)

16. Abstract

General: The development of a computer simulation program system for generating highway traffic crossing a bridge and calculating the resulting stresses and stress ranges is described. The total system consists of four stand-alone programs which can be used either independently or as a dependent series where output from one program is used as input to the next. Output consists of loading and stress range histograms.

Report 73-43: A description of the traffic loading generator program BRIGLDI is given. Included are utilization instructions, data preparation instructions, output and variables definitions, and a sample case.

This report is the second in a series. The others in the series are:

FHWA No.	Short Title
FHWA-RD-73-42	Study Report
FHWA-RD-73-44	User's Manual - Bridge Dynamic Stress Analysis
FHWA-RD-73-45	User's Manual - Stress Histogram Prediction System

17. Key Words	18. Distribution Statement
bridges, stress history, traffic simulation, stress range, and fatigue	No restrictions. This document is available to the public through the National Technical Information Service, Springfield, Virginia 22151.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	191	

Form DOT F 1700.7 (8-69)

PREFACE

The computer program described in this report was developed as a part of a project which was a continuation of work performed by the Kelly Scientific Corporation on Forecasting of Heavy Loading Patterns on Highway Bridges (1). The computer simulation program developed in that work, BRIGLD1, and as modified by Messrs. Wm. Armstrong and S. Smith of the Federal Highways Administration Research Laboratory, formed the basis of the computer program described in this report.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

a = Acceleration of a vehicle.

A = Vehicle ahead in next lane.

C = Critical value of variables.

C_o = A constant in the weight to power acceleration equation.

C_1 = A coefficient in the velocity term of the acceleration equation.

C_2 = A coefficient in the velocity squared term of the acceleration equation.

C_3 = A coefficient in the grade term of the acceleration equation.

d = A subscript denoting.

D = Distance.

F = Vehicle in front of lead vehicle in same lane.

f_1 = A factor used to calculate desired spacing between vehicles.

f_2 = A factor used to calculate desired spacing between vehicles.

G = Gap.

H = Vehicle length.

L = Lead vehicle.

$M(m)$ = Maneuvering vehicle.

O = Original value of variable.

r = Required value of variable.

s = Space.

$T(t)$ = Present variables.

V = Vehicle speed.

X = Vehicle position.

$\text{Tan}\theta$ = Grade slope

ΔT = Increment of integration in time.

INTRODUCTION

The work performed and the results obtained under Contract FH-11-7904 to the Federal Highway Administration, Structures and Applied Mechanics Division, for a study on "Prediction of Loadings on Highway Bridges--Phase II", is described in four separate reports, FHWA-RD-73-42, 43, 44 and 45.

A primary objective of this study was to extend the traffic simulator computer program developed by the Kelly Scientific Corporation to a useable engineering tool capable of generating bridge loads. It was further the purpose of the study to develop a finite element stress analysis program, including dynamic effects due to the live load, which directly interfaced with the load generator, produce several stress histograms for various bridge and traffic configurations, and to develop and implement an alternative method based upon analytic methods rather than traffic simulation.

Report FHWA-RD-73-43 describes the work performed in the study. This includes the background which led to the work performed, a description of the work performed and problems encountered in revising BRIGLD1. The results of sensitivity testing of BRIGLD1, development of the stress program, generated histograms, description of the analytic methods investigation, and the conclusions and recommendations are also included.

Report FHWA-RD-73-44 is the Users Manual which provides utilization instructions, data preparation instructions, output and variables definitions and a description of the BRIGLD1 computer program. The description includes a narrative descriptive section, flow charts and program listings on the main line program and each subroutine. The use of this program for stress range prediction is illustrated in Figure 1.

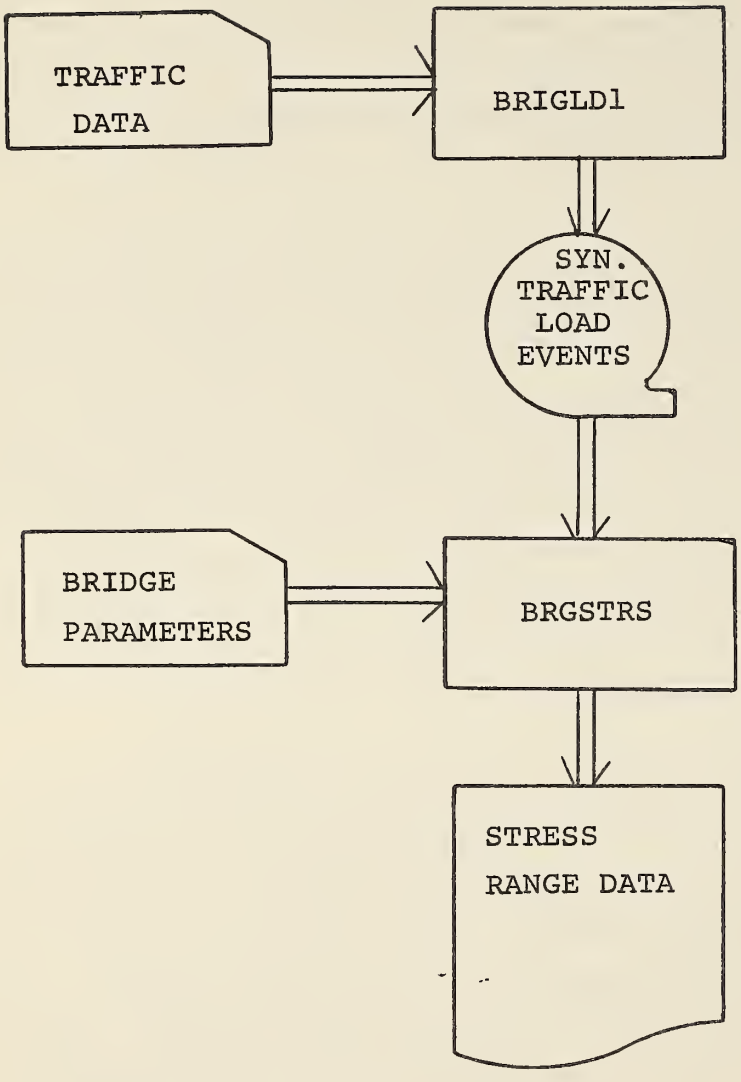


Figure 1. Traffic Simulation Based System

Report FHWA-RD-73-44 is the Users Manual for the dynamic stress analysis computer program, BRGSTRS. This report contains the same descriptive type matter as indicated above for BRIGLD1.

Report FHWA-RD-73-45 contains the Users Manuals for two computer programs which operate as a system with BRGSTRS. The first is the Synthetic Load Generator, SYNGEN, which generates single axle loads for the dynamic stress analysis program, which in turn generates a stress signature curve, trace, for each defined axle load. The second is the histogram computer program, HISGEN, which generates long-term stress range histograms from the synthetic single axle stress trace data generated by the dynamic stress analysis program, as shown in Figure 2. This is accomplished by first forming composite truck stress traces for a given truck population from the single axle data. Then, forming composite truck platoon stress traces for a given platoon population. Long-term effects are estimated from traffic density estimates and the estimated incidence of each platoon configuration. The information contained in this report, for both SYNGEN and HISGEN, is of the same form as described above for BRIGLD1.

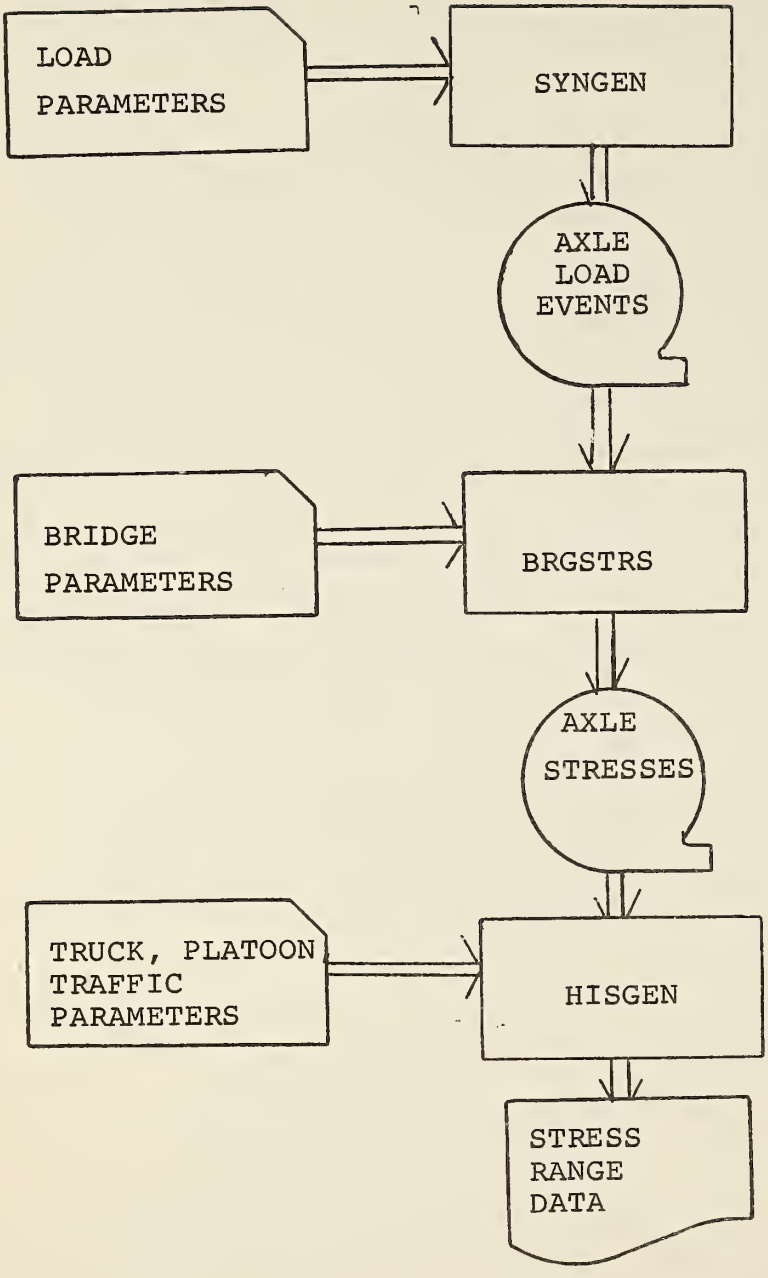


Figure 2. Axle Signature Based System

OPERATING INSTRUCTIONS

Operating instructions for the Bridge Load Generator are divided into four sections, Data Input which includes descriptions of all data, default parameters and data input coding forms, Data Output with examples of printed output and a description of the Bridge Load Data Block, Deck Set Up description, and a Sample Case run.

Data Input

Input data is divided into two sections:

1. Simulation data
2. Subperiod data

The simulation data is read first, followed by as many sets of subperiod data as there are subperiods within the simulation. Data elements, their default values, units and guidelines for data selection follow. Coding forms for data input are also provided.

Simulation Data

All single element data are input under Namelist DATA. Only those elements to be changed from the default values need be entered and the order of entry within the namelist is arbitrary. Data Form 1 may be used for the NAMELIST input (see Figure 3). The first card $DATA and last card $END must be provided even if there is no data to be entered. A comma must follow each data element except the last. The following elements are input under namelist DATA:

Item	Default	Units	Maximum	Description
NTH	1		None	Number of subperiods.
TIMLIM	1	Sec	None	Simulation time limit. This should be equal to the sum of

Item	Default	Units	Maximum	Description
				the subperiod times, however, the simulation will terminate at the shorter of the two times. A continuous simulation time of more than about 5 hours is not recommended unless the program is modified to allow re-seeding of the random number generator.
DELTIM	1.0	Sec	~2.0 Sec	Motion integration interval recommended values are from .5 to 1.0 sec.
MD	11		20	Number of vehicle types.
NL	2		2	Number of lanes. For bi-directional traffic with one lane in each direction, NL=2.
ND	1		2	Number of directions. 1= forward traffic only, 2= bidirectional traffic.
NRAND	00000000			Initializing random number generator seed. A different arbitrary number must be input to provide a different sequence of traffic events.
IOUT	2			Output device number for Bridge Load Data, default is set to punched cards and should be changed to a tape or disk unit.
BRLEN	100°	ft	None	One half of the length of the bridge. BRLEN should be set to the maximum value anticipated for use with the

Item	Default	Units	Maximum	Description
				bridge structural analysis program. Data generated is valid for any bridge equal to or smaller than the specified bridge length.
BRPOS	1100.0	ft	None	Position of the center of the bridge. It is recommended that at least 100 ft be allowed to the starting point of the bridge to provide a traffic distribution in the second lane. Recommended distance to the starting point of the bridge is 1000 ft. 1/2 the bridge length must be added to this value.
NZ	0		5	Number of restricted zones. Used in conjunction with ZONES input. See Figure 2, Data Form 2.
SPDLIM	65.0	mi/hr	None	Car speed limit for the road simulated.
TRKLIM	55.0	mi/hr	None	Truck speed limit for the road simulated.
EXSPD	15.0	mi/hr	None	Excess speed allowed above the car or truck speed limits for maneuvering vehicles.
SPDMIN	40.0	mi/hr	TRKLIM	Minimum speed allowed any vehicle during the simulation. It is recommended that this value not be set above 40 mi/hr. Very heavy traffic will tend to slow down to this value, especially

Item	Default	Units	Maximum	Description
				for longer roadways.
ACCEL	15	ft/sec^2	None	Maximum permitted acceleration. This number should be consistent with current vehicle performance specifications.
SDFAC*	15.0		None	Factor used to calculate minimum desired gap between vehicles.
SAFDIS	10.0	ft	None	Minimum distance permitted between the rear bumper of the lead vehicle and the front bumper of the following vehicle.
LT	12		50	Number of load intervals used to generate load histograms. Interval size is input as TALINC. Default for TALINC is 8000 lb, which produces load intervals of 0-8000 lb 8000-16000 lb 1600-2400 lb ⋮ 88000 to 9600 lb and above 9600 lb
TALINC	8000	lb	None	Load increment for load histograms. See LT above.
DBUG	F	Logical	T	Switch to turn on a Debug printout. It is not recommended to use this set True for runs longer than about 200 sec because of the volume of data that is printed.

*See Sensitivity Analysis Report FHWA-RD-73-

SIMULATION SINGLE PARAMETERS (NAMELIST INPUT)

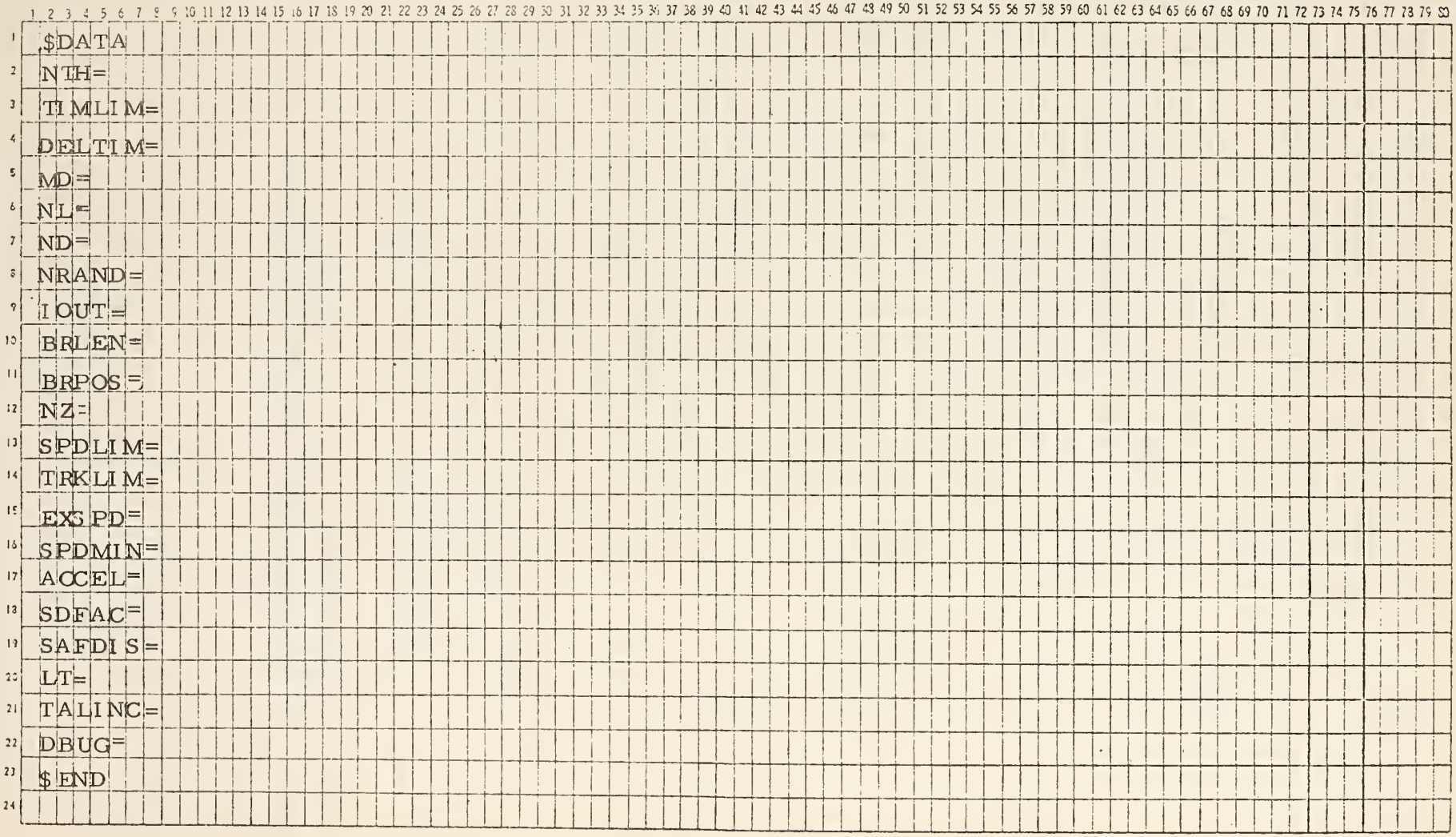


Figure 3. Data Form 1

Tabular data may be entered, or the default values may be used. Tabular data includes vehicle description, Restricted Zone data, and Coefficients of Acceleration. These may be coded using Data Form 2 (see Figure 4). Vehicle Data may be entered by the following input:

Card	Column	Item
1	1-8	"VEHICLES"
1	11-12	Number of Vehicle Types--This number overrides the MD designation.
2-21	1	Number of axles.
2-21	2-6	Vehicle Power
2-21	7-10	Vehicle Length
2-21	11-14	Axle Position - first axle
2-21	15-17	Axle Weight - percentage on first axle
2-21	18-21	Axle Position - second axle
2-21	22-24	Axle Weight - percentage on second axle
2-21	25-28	Axle Position - third axle
2-21	29-31	Axle Weight - percentage on third axle
2-21	32-35	Axle Position - fourth axle
2-21	36-38	Axle Weight - percentage on fourth axle
2-21	39-42	Axle Position - fifth axle
2-21	43-45	Axle Weight - percentage on fifth axle

All double axles (separated by less than five feet) are compressed to single axle values at the midpoint of the axle positions. This data need be entered only if it differs from the default values. Default values are listed below:

Vehicle Type	Number Axles	Power	Vehicle Length	Axle No.	Axle* Position	Axle* Weight
1-car	2	100	19	1	3.0	.50
				2	14.0	.50
2-2D	2	136	23	1	4.0	.25
				2	19.0	.75

Vehicle Type	Number Axles	Power Vehicle Length	Axle No.	Axle* Position	Axle* Weight
3-3	2	157	28	1	4.0	.25
				2	20.0	.75
4-2S1	3	165	38	1	4.0	.20
				2	15.5	.40
				3	32.0	.40
5-2S1	3	165	46	1	4.0	.20
				2	15.5	.40
				3	42.0	.40
6-2S1	3	165	54	1	4.0	.20
				2	15.5	.50
				3	48.0	.30
7-2S2	3	172	46	1	4.0	.20
				2	15.5	.40
				3	38.0	.40
8-2S2	3	172	50	1	4.0	.10
				2	15.5	.40
				3	42.0	.50
9-2S2	3	184	54	1	4.0	.10
				2	15.5	.30
				3	46.0	.60
10-3S2	3	184	46	1	4.0	.20
				2	17.5	.40
				3	38.0	.40
11-3S2	3	184	54	1	4.0	.20
				2	17.5	.40
				3	46.0	.40

*All double axles are shown as single axles at the midpoint.

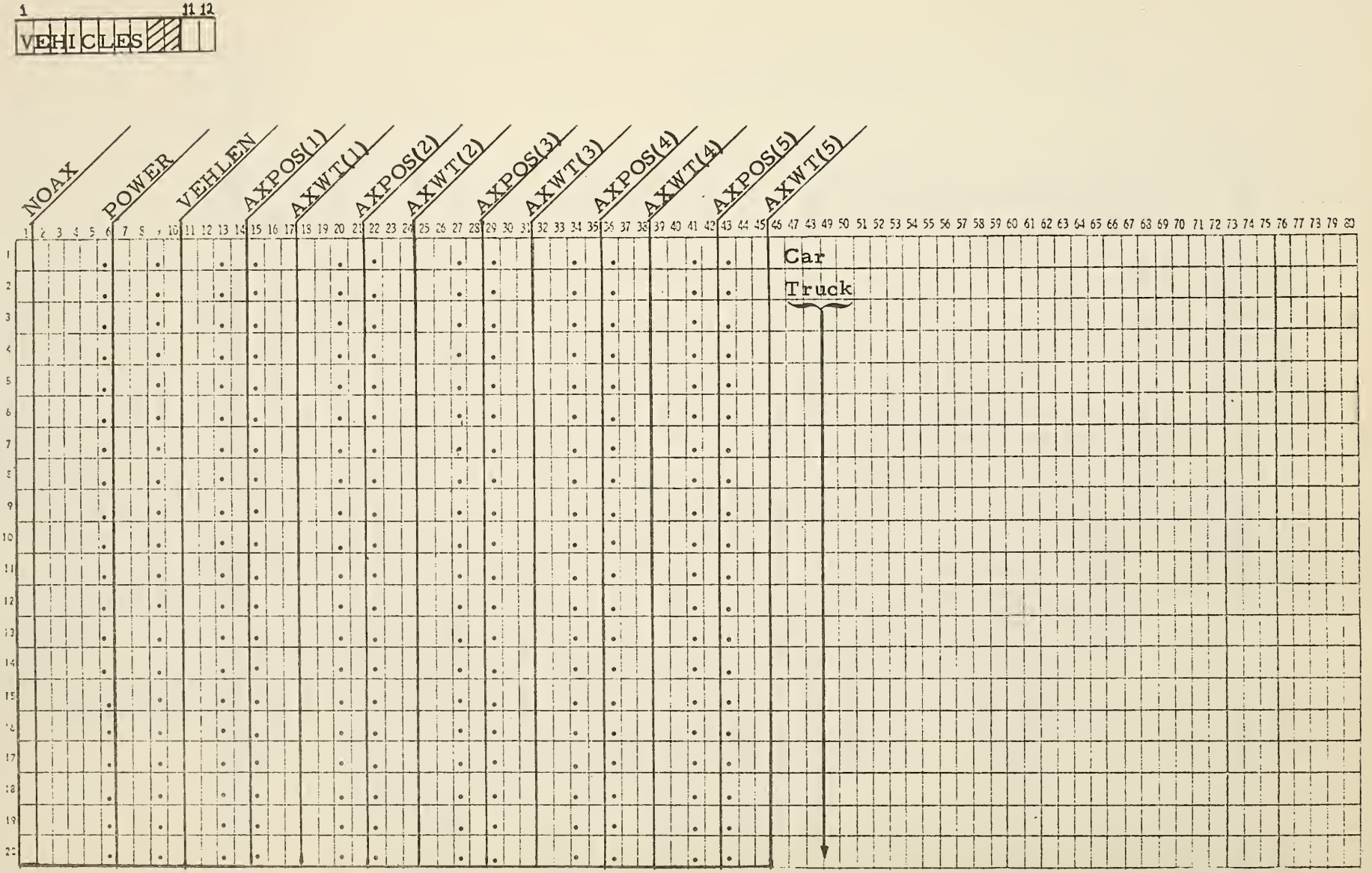
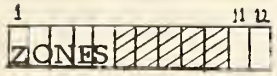
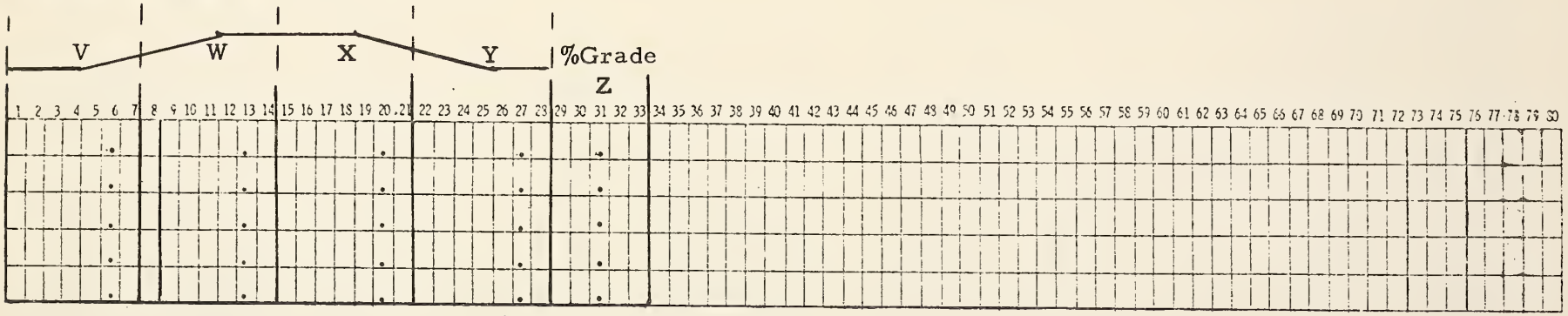
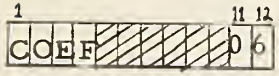


Figure 4. Data Form 2



Restricted Zones





Coefficients of Acceleration $FT(1, I)\ FT(2, I)V\ FT(3, I)V^2\ FT(4, I)\text{TAN}\theta$

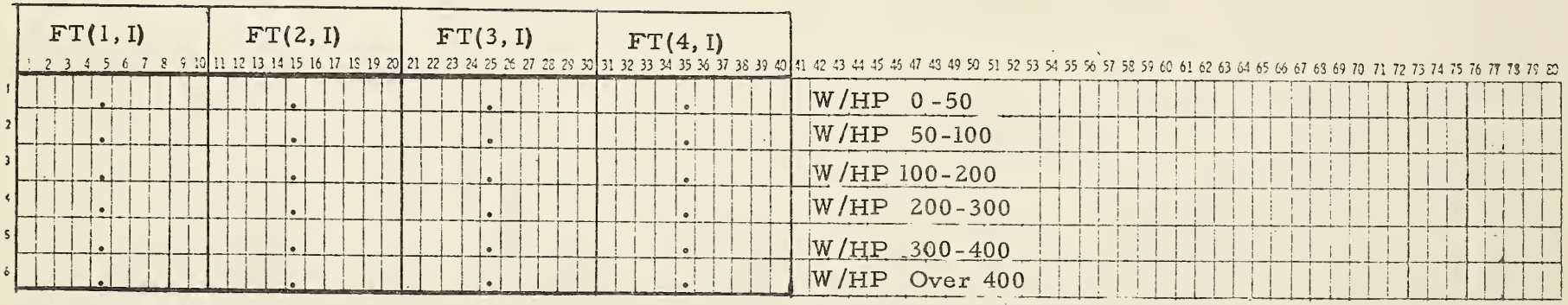


Figure 4. Data Form 2 (Continued)

Restricted zone data may be entered as follows:

Card	Column	Item
1	1-5	"ZONES"
1	12	Number of restricted zones. Maximum = 5, if = 0 this card should not be used
2-6	1-7	V - Beginning of restricted zone, upgrade in forward direction
2-6	8-14	W - End of restricted zone, upgrade in forward direction
2-6	15-21	X - Beginning of restricted zone, downgrade in forward direction
2-6	22-28	Y - End of restricted zone, downgrade in forward direction
2-6	29-33	Z - Percent grade (always positive number) or 0 (indicates a curve or otherwise restricted zone)

Up to five zones may be entered. Four roadway location positions and a grade indicator may be entered for each zone. Each zone may consist of an up and a downgrade or only one of them; or it may consist of one or two curves. A grade indicator of zero implies a curve. The circulation is from the viewpoint of the forward direction of traffic. The first two location positions entered for the zone are the beginning and end of an upgrade; the second two location positions are the end and the beginning of a downgrade. Both grades must be of the same steepness if they are in the same zone but in opposite directions. If an upgrade and a downgrade which differ in the degree of slope are desired, they must be entered as two different zones. Their relative positions, however, are not important. The location of the upgrade may occur after the downgrade. Thus, the first three

road configurations represented in Figure 5 may each be entered as one zone, but the last configuration with different grades requires two zones. The upgrade would be entered as a zone with blanks for the third and fourth positions; while the downgrade would be entered as another zone with blanks for the first and second positions. Default zone values are all zero.

Coefficients of the acceleration equation may be entered as follows:

Card	Column	Item
1	1-4	"COEF"
1	11-12	"06" all six sets of coefficients must be entered
2-7	1-10	FT (1, I) constant coefficient
2-7	11-20	FT (2, I) coefficient of V
2-7	21-30	FT (3, I) coefficient of V^2
2-7	31-40	FT (4, I) coefficient of Tan Θ

The following default values are supplied.

		Coefficients of Acceleration			
I.	Weight/Horsepower	FT(1,I) +	FT(2,I)V +	FT(3,I)V^2 +	FT(4,I)TAN Θ
1	0-50	14.7	0.100	0.0	140.0
2	50-100	11.7	0.090	0.0	120.0
3	100-200	13.0	0.247	0.00118	90.0
4	200-300	9.3	0.198	0.00107	44.0
5	300-400	5.7	0.150	0.00100	28.0
6	Over 400	4.0	0.102	0.00065	38.0

It is not anticipated that the user would want to change these values.

Tabular Data must be followed by an END card--columns 1-3. Whether any tabular data is entered or not, the END card must appear in the data input deck.

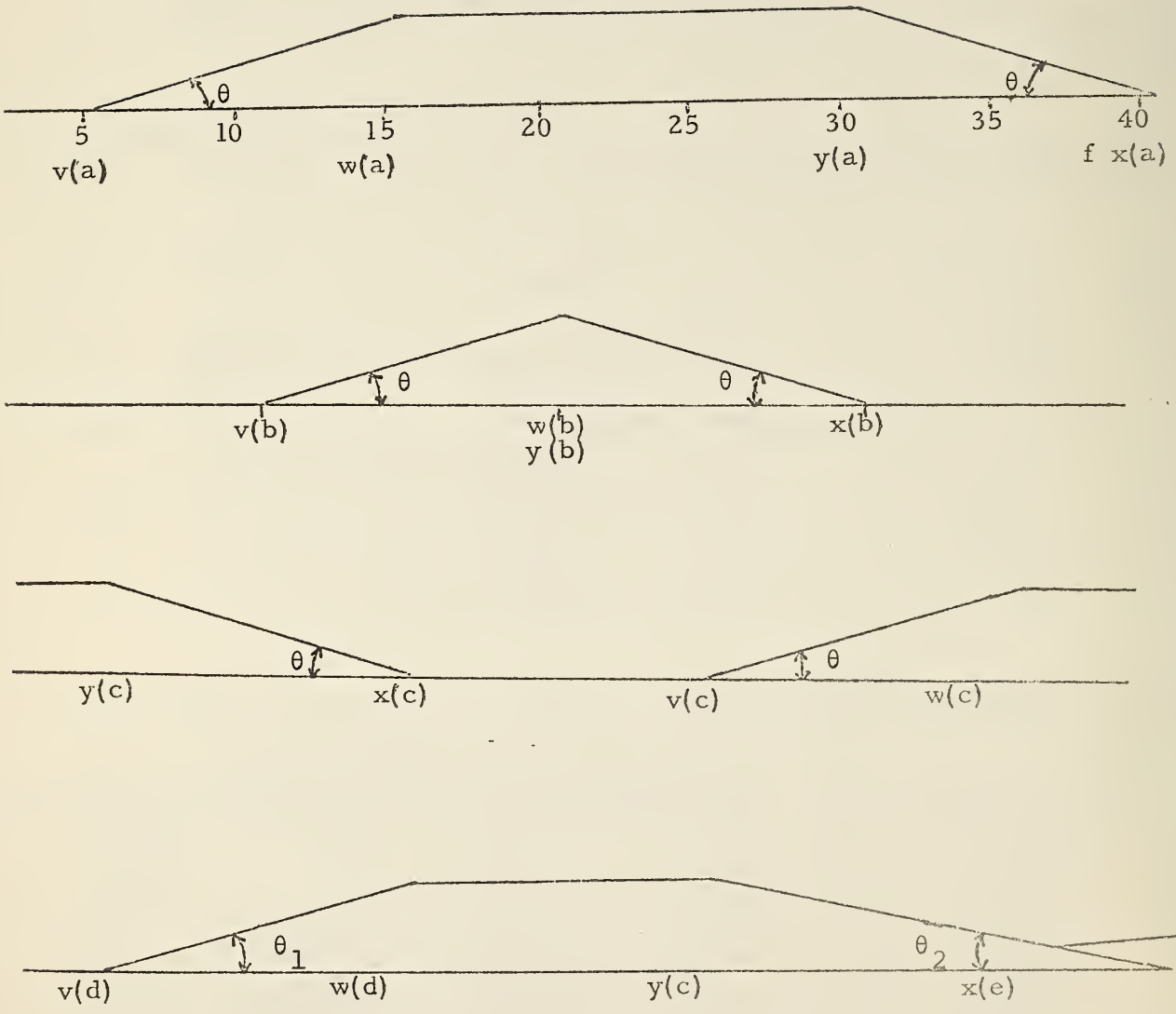


Figure 5. Restricted Zone Configurations

Subperiod Data

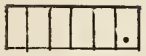
Data which may be changed during the simulation on an elapsed time basis are:

- Vehicle Type Distribution
- Headway Distribution
- Vehicle Speed Distribution
- Vehicle Weight Distribution
- Truck Platoon Distribution

Any or all of the above data may be changed at the beginning of each subperiod. Data Form 3, Figure 6 may be used for subperiod data input. Subperiod data is input as follows:

Card	Columns	Item
1	1-6	Subperiod time in seconds
2	1-4	JTY = number of vehicle type distribution tables to be read (max. = 2)
2	5-8	JHD = number of headway tables to be read (max. = 2)
2	9-12	JSD = number of speed tables to be read (max. = 2)
2	13-16	JWT = number of weight tables to be read (max. = 2)
2	17-20	JPL number of Platoon Distribution tables to be read (max. = 2)

The first two cards must appear whether any subperiod data is to be read or not, following these are only the distribution tables specified by nonzero entries for JTY, JHD, JSD, JWT, and JPL. All distribution table values are entered 10 to a card, 8 columns per value, punched decimal. Except for AFR, for which there must be MD values, the number of values to be read in for each table must be placed in the first 4 columns of a card preceeding the values.



JTY	JHD	JSD	JWT	JPL

VEHICLE TYPE DISTRIBUTION

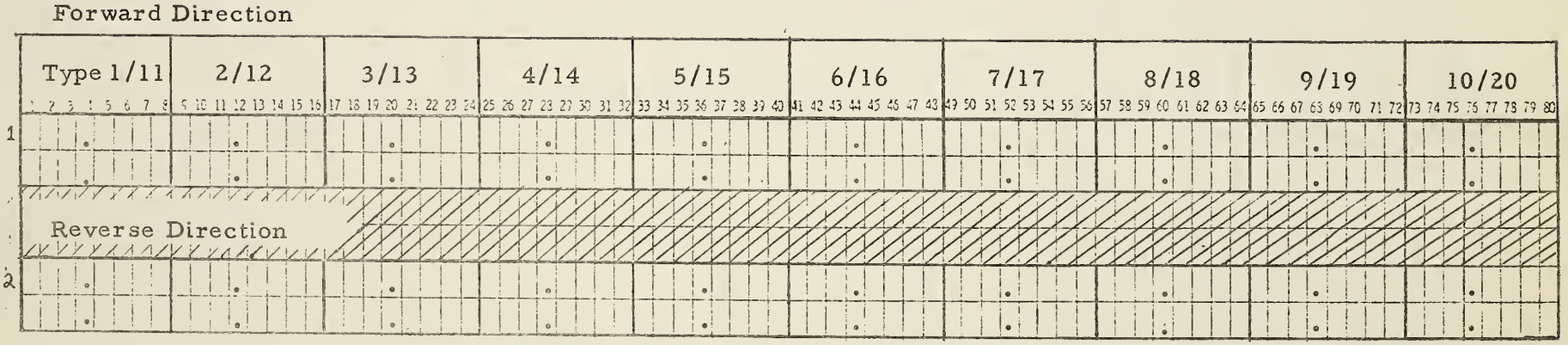


Figure 6. Data Form 3

HDTAB Headway Distribution Tables

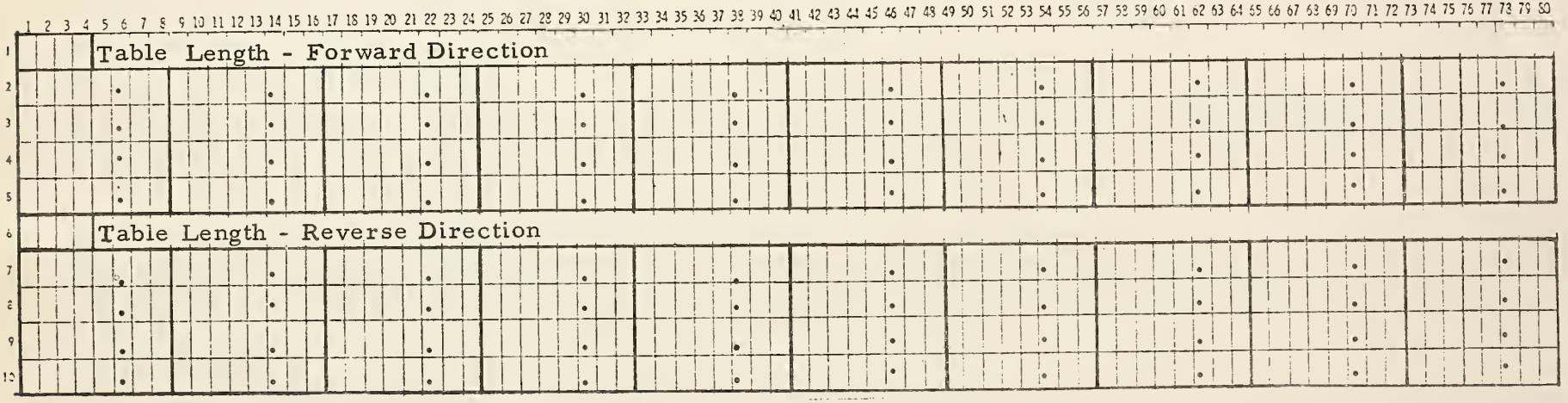


Figure 6. Data Form 3 (Continued)

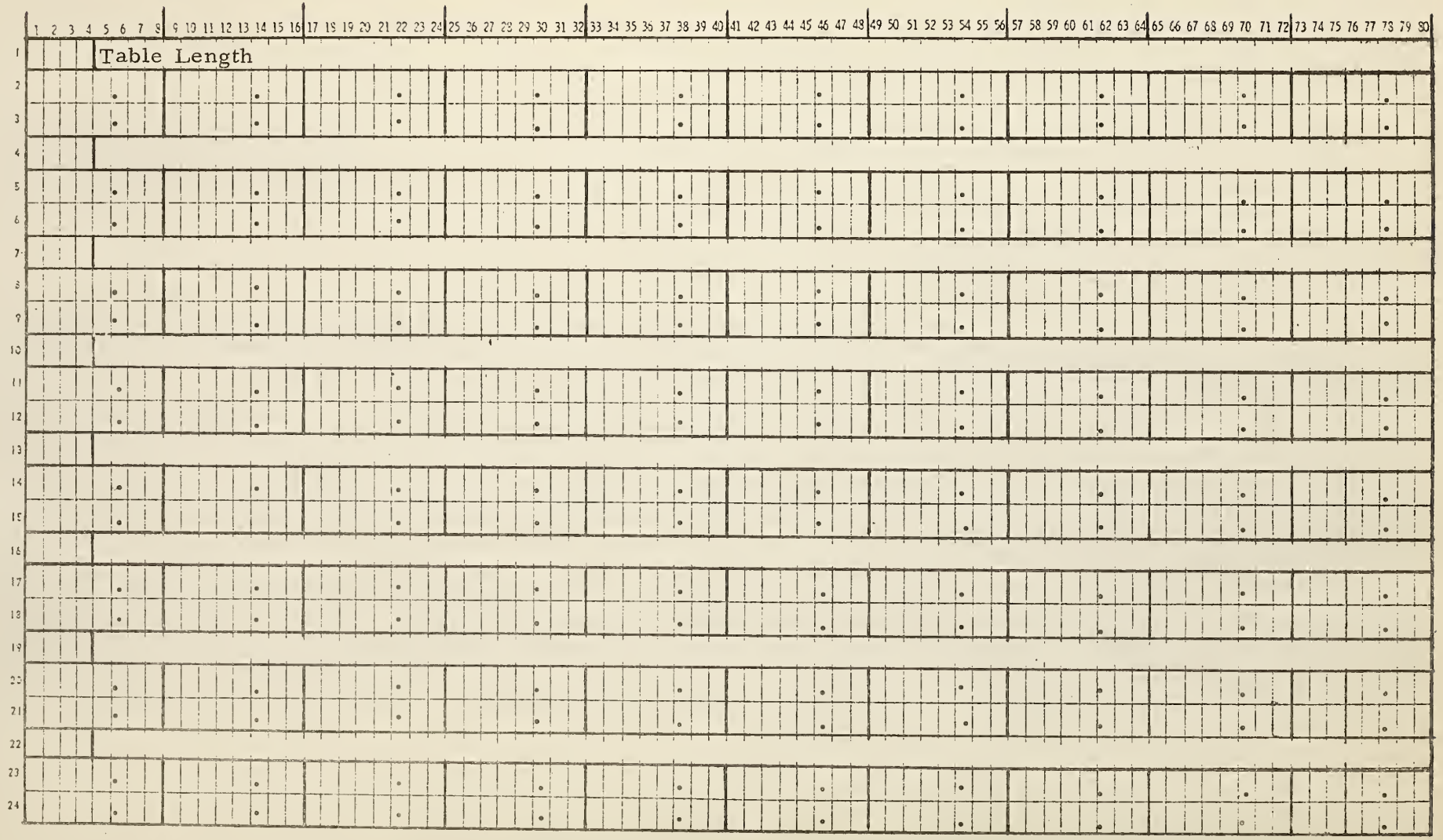


Figure 6. Data Form 3 (Continued)

WTAB Weight Distribution Tables

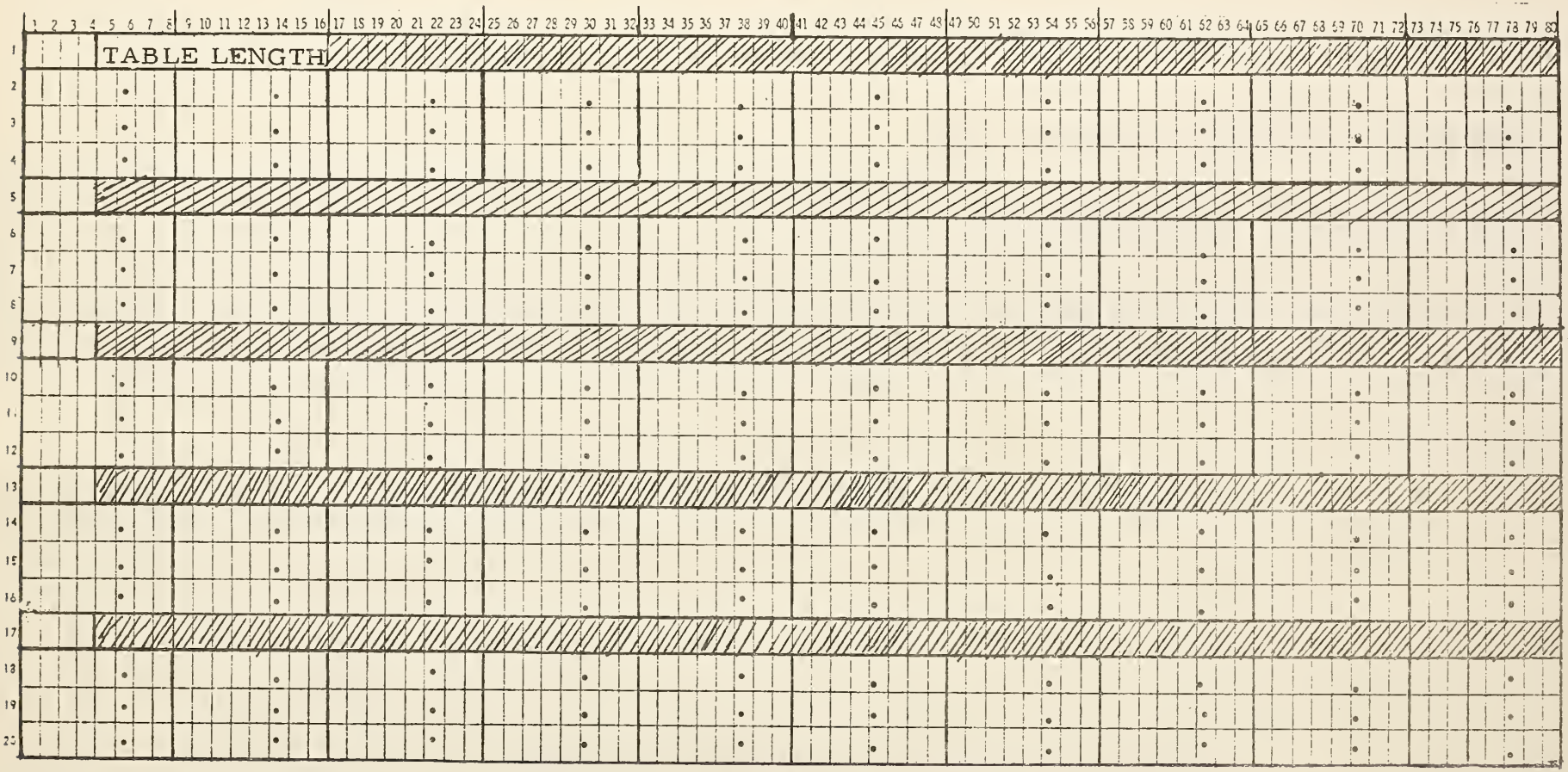


Figure 6. Data Form 3 (Continued)

DPLTON Truck Platoon Distribution Tables

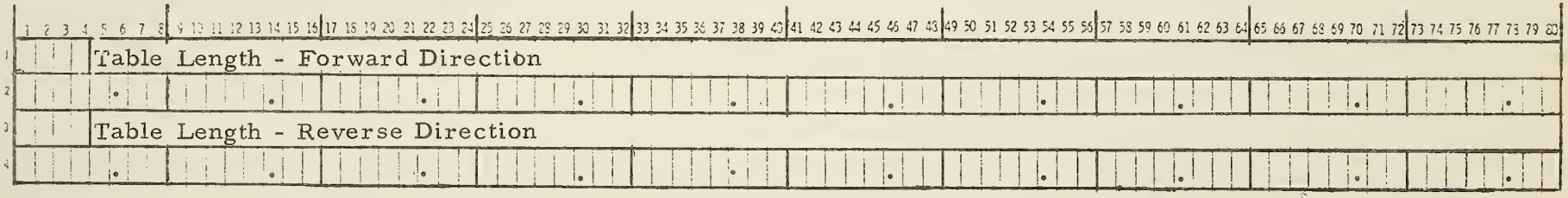


Figure 6. Data Form 3 (Continued)

1. AFR - vehicle type distribution table, JTY (max. = 2) tables must be entered, each table consists of :

Fraction of each vehicle type entered in fields of 8 columns per value for each type.

Default Values are:

Vehicle Type	Forward	Reverse
1	.83	.83
2	.875	.875
3	.885	.885
4	.888	.888
5	.891	.891
6	.894	.894
7	.906	.906
8	.919	.919
9	.953	.953
10	.988	.988
11	1.000	1.000
12	1.000	1.000
13	1.000	1.000
14	1.000	1.000
15	1.000	1.000
16	1.000	1.000
17	1.000	1.000
18	1.000	1.000
19	1.000	1.000
20	1.000	1.000

2. HDTAB - headway tables, JHD (max. = 2) tables must be entered, each table consists of:

   a. First card, NYMD in columns 1-4, number of table values, right-justified, no decimal
   b. Next cards, NUMD uniformly distributed headway values.

Default HDTAB values are:

	Forward	Reverse
Table Length	21	21
Values		
1	.40	.40
2	.50	.50
3	.60	.60
4	.70	.70
5	.80	.80
6	.90	.90
7	1.00	1.00
8	1.10	1.10
9	1.30	1.30
10	1.50	1.50
11	1.60	1.60
12	1.80	1.80
13	2.00	2.00
14	2.10	2.10
15	2.50	2.50
16	2.80	2.80
17	3.00	3.00
18	3.50	3.50
19	4.10	4.10
20	5.20	5.20

3. SDTAB - speed tables, JSD (max. = 20) tables must be entered.

a. Uniformly distrubuted speed input per vehicle type is similar to headway tables.

b. If JSD ≠ MD, all remaining vehicle types will use values from last table entered.

Default length on all Speed Tables is 11. The default SDTAB values are:

Vehicle Type Value	1	2	3	4	5	6	7	8	9	10	11
1	40	40	30	32	32	32	32	32	32	32	32
2	66	62	56	60	60	60	60	60	60	60	60
3	71	66	60	66	66	66	66	66	66	66	66
4	74	69	64	70	70	70	70	70	70	70	70
5	77	71	67	73	73	73	73	73	73	73	73
6	80	74	70	76	76	76	76	76	76	76	76
7	83	76	72	79	79	79	79	79	79	79	79
8	86	78	75	82	82	82	82	82	82	82	82
9	89	81	78	86	86	86	86	86	86	86	86
10	94	85	83	91	91	91	91	91	91	91	91
11	120	107	109	120	120	120	120	120	120	120	120

4. WTAB - weight tables, JWT(max. = 2) tables must be entered.

a. Uniformly distributed weight input is sililar to headway tables.

b. A single weight distribution must be entered as a two value table, see the default values for Type 1 as an example.

Default WTAB values are:

Vehicle Type	1	2	3	4	5	6
Table Length	2	21	22	21	21	21
Values						
1	3000	3000	12000	12000	12000	12000
2	3000	6500	16000	16900	16900	16900
3	0	7300	17600	18500	18500	18500
4	0	8000	19200	19400	19400	19400
5	0	8700	20900	20000	20000	20000
6	0	9800	22600	20200	20200	20200
7	0	10000	25050	21500	21500	21500
8	0	10400	27500	22400	22400	22400
9	0	11000	29750	23400	23400	23400
10	0	11800	32000	24800	24800	24800
11	0	12500	33250	26800	26800	26800
12	0	13400	34500	29300	29300	29300
13	0	14500	35300	30800	30800	30800
14	0	16000	36100	31600	31600	31600
15	0	17800	37050	32300	32300	32300
16	0	18800	38000	33300	33300	33300
17	0	19800	39550	34100	34100	34100
18	0	20800	41100	35300	35300	35300
19	0	21300	46650	37000	37000	37000
20	0	22600	52200	39600	39600	39600
21	0	25000	55150	59700	59700	59700
22	0	0	58100	0	0	0

Default WTAB values are continued on the next page.

Vehicle Type	7	8	9	10	11
Table Length	21	21	21	21	21
Values					
1	15000	15000	15000	21000	21000
2	19800	19800	19800	23000	23000
3	21300	21300	21300	25000	25000
4	22700	22700	22700	26000	26000
5	24000	24000	24000	27000	27000
6	25800	25800	25800	29500	29500
7	28500	28500	28500	34700	34700
8	32000	32000	32000	45200	45200
9	34400	34400	34400	52000	52000
10	36700	36700	36700	56800	56800
11	38900	38900	38900	60000	60000
12	41700	41700	41700	62700	62700
13	44600	44600	44600	65000	65000
14	47100	47100	47100	66300	66300
15	48800	48800	48800	67300	67300
16	52600	52600	52600	68200	68200
17	55000	55000	55000	69700	69700
18	57000	57000	57000	70200	70200
19	58800	58800	58800	71000	71000
20	61000	61000	61000	71500	71500
21	65900	65900	65900	83900	83900
22	0	0	0	0	0

5. DPLTON Truck Platoon Distribution tables, JPL (max. = 2) tables must be entered.

   a. First card NUMD in columns 1-4 number of table values

   b. NUMD table values (max. = 10) percentage occurrence of platoon size 1, 2, 3 . . . NUMD.

Default Platoon Distribution is all single truck events, that is, no platoons.

Data Output

Output from the Bridge Load Generator consists of:

Simulation Run Input Parameters

Bridge Load Data Block

Simulation Run Statistics

Simulation Run Input Parameters - Simulation Run Parameter printout (see Figure 7, Simulation Run Parameters) consists of the NAMELIST DATA printout, Restricted Zone data, Vehicle Specification data, Coefficients of Acceleration default values or input data. Note that speed parameters SPDLIM, TRKLIM, EXSPD, and SPDMIN have been converted from mi/hr to ft/sec. If no values are input, the program default values are printed. This is followed by a printout of the default values for the Subperiod data, that is, Traffic Distribution, Truck Platoon Distribution, Headway Tables, Weight Tables, and Speed Tables (see Figure 8, Simulation Subperiod Parameters). In addition, at the beginning of the simulation run the following is printed out:

"A SIMULATION TO REPRESENT A PERIOD OF _____ HOURS. VEHICLES ARE GENERATED _____ FEET FROM BRIDGE CENTER. WEIGHTS ON BRIDGE ARE SUMMED AND COUNTED FOR LOAD INCREMENTS OF _____ POUNDS UP TO _____. _____ PERIOD TYPES AND _____ VEHICLE TYPES ARE CONSIDERED."

If, at the beginning of any subperiod any of the Traffic Distribution, Headway Distribution, Speed Distribution, Weight Distribution or Platoon Distribution Tables are changed, the corresponding tables are printed as changed. Bridge Load Data blocks are written out as they are generated during the simulation. These blocks may go out to tape or disk, cards may be used for extremely short runs as 19 cards are produced for each block of data, every Δt when there is a truck on the deck. The Bridge Load Data block consists of the following elements:

Item	Dimension	Description
SUMHRX	1	Simulation time
DTL	1	Integration interval
IEVNTX	1	Event number
NOAXLX	1	Number of axle loads on-going on the bridge
NTRUKX	1	Number of trucks this event
LAST	1	Logical = True if this is the last block of this event

Item	Dimension	Description
LTYPE	20	Truck type
WGTX	20	Weight of truck
SPDX	20	Speed of truck
LLANE	20	Lane of truck
TIMEX	20	Time truck went on the bridge
XPOSX	50	Axle position
LNUMX	50	Axle lane
WEITX	50	Axle weight
DXPOSX	50	Distance axle will travel during integration interval
ACCLRX	50	Acceleration of axle

The truck identification data,LTYPE, WGTX, SPDX, LLANE, and TIMEX, is captured as a truck goes on the bridge and does not change during an event except that new trucks are added to the list as they occur during the event. The axle load data, XPOSX, LNUMX, WEITX, DXPOSX, and ACCLRX are current values for each integration step.

Simulation Run Statistics are output every hour of the simulation and at the end of a simulation run. They are composed of:

- Total Vehicles Generated
- Simulated Time - starting from occurrence of first vehicle going on the bridge
- Platoon Distributions - collected at time of generation--forward direction and reverse direction (if ND = 2), and sampled on the bridge
- Vehicle Type Distribution
- Load Distribution

A sample statistical output is shown in the section - Sample Case below.

Deck Set Up

The program deck should be set up as follows:

Users job card

JCL specifying program source and/or program source deck or object deck or combination of both

Data output specification - JCL consisting of a card specifying printer output on device 06 and a card specifying tape output on device came number as IOUT tape or permanent disk file

Data input specification - JCL consisting of a card specifying data deck input on device 05

Data Deck set up as described in section on data input above

Figure 7, Deck Set Up, further illustrates the above. An example is found below under the section Sample Case.

The following running times are given as guidelines in specifying running time for the job. Variations in some parameters cause variations in running time. The most significant are given below:

Roadway & Bridge Length	Δt	Restricted Zones	Number Directions	Simulated Time	CPY Time
1200 ft.	1 sec.	0	1	1 hr.	21 sec.
1200 ft.	1 sec.	5	1	1 hr.	85 sec.
2000 ft.	1 sec.	0	2	1 hr.	600 sec.

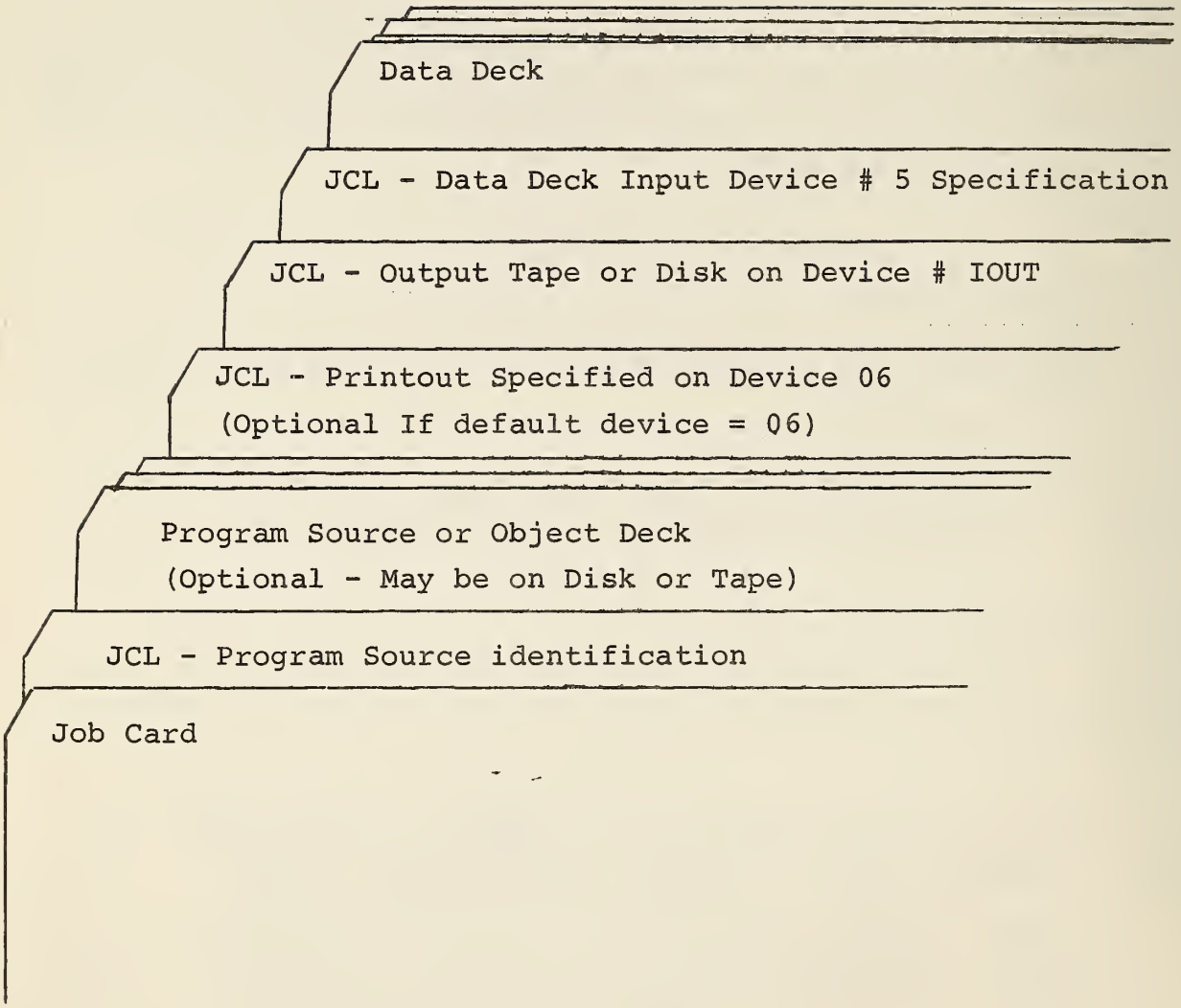


Figure 7. Deck Set Up

Sample Case

The following sample case is provided here as an example for the user. It is not recommended that the source be recompiled for each run. A program object should be stored on disk if frequent use is to be made of the program. An "off-line" listing is provided of the cards, JCL, and data, used to run the program (see Figure 8). An "on-line" program printout is provided as Figure 9.

```
//R4111PF3 JOB (0118,FHRS),'PAT FOTHERGILL',TIME=2,CLASS=M,REGION=200K
//CL EXEC FORTGCLG,PARM.FORT=(SOURCE,MAP),TIME=2
//FORT.SYSIN DD *
```

[Program Source Deck Goes Here]

```
//GO.SYSUDUMP DD DUMMY,DISP=(NEW,DELETE)
//GO.FT06F001 DD SYSOUT=A,DCB=(RECFM=UA,BLKSIZE=133)
//GO.FT07F001 DD DSN=FOTHGL,UNIT=2400-4,VOL=SER=7967,
//               DISP=(NEW,KEEP),LABEL=(1,SL),
//               DCB=(RECFM=VBS,LRECL=1692,BLKSIZE=6772)
//GO.FT05F001 DD *
 &DATA
 TIMLIM=60.0,
 IOUT=07,
 DBUG=.TRUE.,
 &END
END
  60.
   0   0   0   0   1
   6
  .50     .25     .125    .0625   .0313   .0312
/*
//
```

Figure 8. Sample Case Input

```
&DATA
NTH=          1,TIMLIM=  60.000000.    ,DELTIM=  1.0000000    ,MD=        11,NL=          2,ND=          1,NRAND=          0,IOUT=
         2,BRLEN=  100.00000    ,BRPOS=  1100.0000    ,NZ=          0,SPDLIM=  95.333481    ,TRKLIM=  80.656794    ,EXSPD=
   22.000031    ,SPDMIN=  58.666763    ,ACCEL=  15.000000    ,SDFAC=  15.000000    ,SAFDIS=  10.000000    ,LT=          12,TALINC=
   8000.0000    ,DBUG=T
&END
```

ZONE DATA

BEGIN FORWARD UPGRADE	END FORWARD UPGRADE	BEGIN REVERSE UPGRADE	END REVERSE UPGRADE	PERCENT OF GRADE
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0

Figure 9. Sample Case Output

VEHICLE DATA

VEHICLE TYPE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
NUMBER OF AXLES	2	2	2	3	3	3	3	3	3	3	3	0	0	0	0	0	0	0	0	0
VEHICLE POWER	100.	136.	157.	165.	165.	165.	172.	172.	184.	184.	184.	0.	0.	0.	0.	0.	0.	0.	0.	0.
VEHICLE LENGTH	19.0	23.0	28.0	38.0	46.0	54.0	46.0	50.0	54.0	46.0	54.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
FIRST AXLE POSITION	3.0	4.0	4.0	4.0	4.0	4.0	4.0	4.0	4.0	4.0	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PERCENT WEIGHT ON AXLE	.50	.25	.25	.20	.20	.20	.20	.10	.10	.20	.20	.0	.0	.0	.0	.0	.0	.0	.0	.0
SECOND AXLE POSITION	14.0	19.0	20.0	15.5	15.5	15.5	15.5	15.5	15.5	17.5	17.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PERCENT WEIGHT ON AXLE	.50	.75	.75	.40	.40	.50	.40	.40	.30	.40	.40	.0	.0	.0	.0	.0	.0	.0	.0	.0
THIRD AXLE POSITION	0.0	0.0	0.0	32.0	42.0	48.0	38.0	42.0	46.0	39.0	46.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PERCENT WEIGHT ON AXLE	.0	.0	.0	.40	.40	.30	.40	.50	.60	.40	.40	.0	.0	.0	.0	.0	.0	.0	.0	.0
FOURTH AXLE POSITION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PERCENT WEIGHT ON AXLE	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0
FIFTH AXLE POSITION	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
PERCENT WEIGHT ON AXLE	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0

Figure 9. Sample Case Output (Continued)

WEIGHT TABLES

VEHICLE TYPE VALUE	1	2	3	4	5	6	7	8	9	10	11	12
1	3000.	3000.	12000.	12000.	12000.	12000.	15000.	15000.	15000.	21000.	21000.	0.
2	3000.	6500.	16000.	16900.	16900.	16900.	19800.	19900.	19900.	23000.	23000.	0.
3	0.	7300.	17600.	18500.	18500.	18500.	21300.	21300.	21300.	25000.	25000.	0.
4	0.	8300.	19200.	19400.	19400.	19400.	22700.	22700.	22700.	26000.	26000.	0.
5	0.	8700.	20900.	20000.	20000.	20000.	24000.	24000.	24000.	27000.	27000.	0.
6	0.	9600.	22600.	20200.	20200.	20200.	25800.	25800.	25800.	29500.	29500.	0.
7	0.	10000.	25050.	21500.	21500.	21500.	28500.	28500.	28500.	34700.	34700.	0.
8	0.	10400.	27500.	22400.	22400.	22400.	32000.	32000.	32000.	45200.	45200.	0.
9	0.	11000.	29750.	23400.	23400.	23400.	34400.	34400.	34400.	52000.	52000.	0.
10	0.	11800.	32000.	24800.	24800.	24800.	36700.	36700.	36700.	56800.	56800.	0.
11	0.	12500.	33250.	26800.	26800.	26800.	38900.	38900.	38900.	60000.	60000.	0.
12	0.	13400.	34500.	29300.	29300.	29300.	41700.	41700.	41700.	62700.	62700.	0.
13	0.	14500.	35300.	30800.	30800.	30800.	44600.	44600.	44600.	65000.	65000.	0.
14	0.	16000.	36100.	31600.	31600.	31600.	47100.	47100.	47100.	66300.	66300.	0.
15	0.	17000.	37050.	32300.	32300.	32300.	48800.	48800.	48800.	67300.	67300.	0.
16	0.	18000.	38000.	33300.	33300.	33300.	52600.	52600.	52600.	68200.	68200.	0.
17	0.	19600.	39550.	34100.	34100.	34100.	55000.	55000.	55000.	69700.	69700.	0.
18	0.	20800.	41100.	35300.	35300.	35300.	57000.	57000.	57000.	70200.	70200.	0.
19	0.	21300.	46050.	37000.	37000.	37000.	58800.	58800.	58800.	71000.	71000.	0.
20	0.	22600.	52200.	39600.	39600.	39600.	61000.	61000.	61000.	71500.	71500.	0.
21	0.	25000.	55150.	59700.	59700.	59700.	65900.	65900.	65900.	83900.	83900.	0.
22	0.	0.	58100.	0.	0.	0.	0.	0.	0.	0.	0.	0.
23	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
24	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
25	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
26	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
27	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
28	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
29	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
30	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

Figure 9. Sample Case Output (Continued)

SPEED TABLES

VEHICLE TYPE VALUE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	40.	40.	30.	32.	32.	32.	32.	32.	32.	32.	32.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2	60.	62.	56.	60.	60.	60.	60.	60.	60.	60.	60.	0.	0.	0.	0.	0.	0.	0.	0.	0.
3	71.	66.	60.	66.	66.	66.	66.	66.	66.	66.	66.	0.	0.	0.	0.	0.	0.	0.	0.	0.
4	74.	69.	64.	70.	70.	70.	70.	70.	70.	70.	70.	0.	0.	0.	0.	0.	0.	0.	0.	0.
5	77.	71.	67.	73.	73.	73.	73.	73.	73.	73.	73.	0.	0.	0.	0.	0.	0.	0.	0.	0.
6	80.	74.	70.	76.	76.	76.	76.	76.	76.	76.	76.	0.	0.	0.	0.	0.	0.	0.	0.	0.
7	83.	76.	72.	79.	79.	79.	79.	79.	79.	79.	79.	0.	0.	0.	0.	0.	0.	0.	0.	0.
8	86.	78.	75.	82.	82.	82.	82.	82.	82.	82.	82.	0.	0.	0.	0.	0.	0.	0.	0.	0.
9	89.	81.	78.	86.	86.	86.	86.	86.	86.	86.	86.	0.	0.	0.	0.	0.	0.	0.	0.	0.
10	94.	85.	83.	91.	91.	91.	91.	91.	91.	91.	91.	0.	0.	0.	0.	0.	0.	0.	0.	0.
11	120.	107.	109.	120.	120.	120.	120.	120.	120.	120.	120.	0.	0.	0.	0.	0.	0.	0.	0.	0.
12	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
13	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
14	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
15	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
16	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
17	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
18	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
19	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
20	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

Figure 9. Sample Case Output (Continued)

TRAFFIC DISTRIBUTION

VEHICLE TYPE / DIRECTION	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.830	0.875	0.885	0.883	0.891	0.894	0.906	0.919	0.953	0.988	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	0.830	0.875	0.885	0.883	0.891	0.894	0.906	0.919	0.953	0.988	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

TRUCK PLATOON DISTRIBUTION

NUMBER OF TRUCKS / DIRECTION	1	2	3	4	5	6	7	8	9	10
1	1.000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	1.000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

HEADWAY TABLES

VALUE NUMBER / DIRECTION	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.40	0.50	0.60	0.70	0.80	0.90	1.00	1.10	1.30	1.50	1.60	1.80	2.00	2.10	2.50	2.80	3.00	3.50	4.10	5.20
2	0.40	0.50	0.60	0.70	0.80	0.90	1.00	1.10	1.30	1.50	1.60	1.80	2.00	2.10	2.50	2.80	3.00	3.50	4.10	5.20

VALUE NUMBER / DIRECTION	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
1	5.60	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	5.60	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 9. Sample Case Output (Continued)

COEFFICIENTS OF ACCELERATION

WEIGHT/HORSEPOWER	C(0) +	C(1)V +	C(2)V**2 +	C(3)TAN(THETA)
0-50	14.70000	0.10000	0.0	140.00000
50-100	11.70000	0.09000	0.0	120.00000
100-200	13.00000	0.24700	0.00118	90.00000
200-300	9.30000	0.19800	0.00107	44.00000
300-400	5.70000	0.15000	0.00100	28.00000
OVER 400	4.00000	0.10200	0.00065	38.00000

A SIMULATION TO REPRESENT A PERIOD OF 0.0 HOURS.

VEHICLES ARE GENERATED 1100. FEET FROM BRIDGE-CENTER. WEIGHTS ON BRIDGE ARE SUMMED AND COUNTED FOR LOAD INCREMENTS OF 8000. POUNDS UP TO 88000..
1 PERIOD TYPES AND 11 VEHICLES TYPES ARE CONSIDERED.
SUBPERIOD= 0.60E 02 SECONDS UPDATE PASS EVERY 1.00

TRAFFIC DISTRIBUTION

VEHICLE TYPE / DIRECTION	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.830	0.875	0.885	0.888	0.891	0.894	0.906	0.919	0.953	0.988	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
2	0.830	0.875	0.885	0.888	0.891	0.894	0.906	0.919	0.953	0.988	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

TRUCK PLATOON DISTRIBUTION

NUMBER OF TRUCKS / DIRECTION	1	2	3	4	5	6	7	8	9	10
1	0.500	0.250	0.125	0.063	0.031	0.031	0.0	0.0	0.0	0.0
2	1.000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 9. Sample Case Output (Continued)

```
SIMULATION TIME =      26.00

RCADWAY 0 TO 3000 FT
 --------------------------------------------------------------------------------------------------------------------------
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 0 0 3 0 0 5 0 0 0 0 6 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 --------------------------------------------------------------------------------------------------------------------------
 VEHICLES NO.     6   HEADWAY      2.294 TYPE      1 SPEED     58.667 WEIGHT  3000.000 LENGTH    19.000

EVENT NO.      1   BRIDGE LOAD TIME =      28.0000SEC

          LANE      WEIGHT  POSITION  DISTANCE  ACCELERATION

        1     2      5939.    -25.83     90.74          0.0
        2     2     14843.    -37.38     90.74          0.0
        3     2      8909.    -69.68     90.74          0.0

TRUCKS THIS EVENT      1

TRUCK   TYPE     WEIGHT    SPEED   LANE   TIME ON BRIDGE
    1      5     29695.      91.      2     28.0

BRIDGE LOAD TIME =       28.0000SEC

          LANE      WEIGHT  POSITION  DISTANCE  ACCELERATION  ORDER

        1     2      8909.    -69.68     90.74          0.0
        2     2     14848.    -37.38     90.74          0.0
        3     2      5939.    -25.88     90.74          0.0

EVENT NUMBER    1          NO. OF TRUCKS    1

TYPE     WEIGHT    SPEED     LANE     TIME ENTERING BRIDGE
   6     29695.      91.        2       28.

     I  IFWD  IBAK  INDX  LANE  POSITION    SPEED 1    SPEED 2       ACC   STAT   TYPE
     1     4     6     0     1   1251.68      58.67      58.67       0.0      0      1
     2     3     7     6     2   1159.61      90.74      90.74       0.0      0      6
     3     6     2     5     1    271.51      81.35      81.35       0.0      0      1
     4     7     1     2     1   1047.19      71.42      71.42       0.0      0      3
     5     6     6     1     1    395.74      73.85      73.85       0.0      0      1
     6     5     3     3     1    652.13      72.86      72.86       0.0      0      1
     7     2     4     4     1   1280.25      89.16      89.16       0.0      0      1
```

Figure 9. Sample Case Output (Continued)

TOTAL VEHICLES GENERATED = 33 SIMULATED TIME = 60. SECONDS

PLATOON DISTRIBUTION

	1	2	3	4	5	6	7	8	9	10
GENERATED FORWARD	1	2	0	0	0	0	0	0	0	0
SAMPLED ON BRIDGE	1	1	0	0	0	0	0	0	0	0

TYPE DISTRIBUTION

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
24	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

LOAD DISTRIBUTION

0. TO 8000.	0
8000. TO 16000.	0
16000. TO 24000.	1
24000. TO 32000.	1
32000. TO 40000.	0
40000. TO 48000.	0
48000. TO 56000.	1
56000. TO 64000.	0
64000. TO 72000.	0
72000. TO 80000.	0
80000. TO 88000.	0
88000. TO 96000.	0
ABOVE 96000.	0

Figure 9. Sample Case Output (Continued)

PROGRAM DESCRIPTION

Introduction

The Bridge Load Generator Program is described here in terms of its individual subroutines and data interfaces. The section on each subroutine will contain a description of the routine, the program flow chart, and a program listing. Preceeding the subroutine descriptions is a list of program variables. This list contains all variables that are used by more than one subroutine or are used for program data input. Units or data type associated with the variable are provided where applicable, table dimensions, and a list of subroutines using each variable is included.

List of Variables

The program List of Variables, Table 1, defines all data elements and variables which provide interfaces between program subroutines or those data elements which are required for data input or output. Consequently, program data interfaces will not be restated in the Program Routines section.

TABLE 1. LIST OF VARIABLES

SYMBOL USED IN PROGRAM	DIMENSION	UNITS/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
ACC	(400)	ft/sec^2	Acceleration associated with vehicle.	CALACC GEN PASPOS PASTES SORPOS UPDATE
ACCEL		ft/sec^2	Maximum permitted acceleration.	CALACC INDATA UPDATE
ACCLR	(50)	ft/sec^2	Bridge Loading Data--unordered acceleration of axle on deck.	ORDER UPDATE
ACCLRX	(50)	ft/sec^2	Bridge Loading Data--ordered output block acceleration of axle on deck.	ORDER
AFR	(20, 2)		Vehicle type distribution. Fraction of traffic representing each type in each direction. This table is modified to maintain correct distributions if Truck Platoon distributions are specified.	GEN
AFS	(20, 2)		Same as AFR not modified for Platoon distributions.	INDATA READ
AXPOS	(5, 20)	ft	Axle position from front bumper by vehicle type. This table is modified from input to compress double axles into one axle at the midpoint.	INDATA

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNITS/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
AXWT	(5, 20)	%	Percent weight on each axle. This table is modified from input to compress double axles.	INDATA UPDATE
BREND		ft	Position of the end of the bridge. BREND BRPOS BRLEN	BRGLOD UPDATE
BRLEN		ft	Half of bridge length.	BRGLOD INDATA
BRPOS		ft	Center of the bridge relative to the beginning of the roadway.	BRGLOD INDATA REZONE
BRST		ft	Position of the beginning of the bridge BRST BRPOS - BRLEN	BRGLOD CONTRO UPDATE
CRIGAP		ft	Minimum gap acceptable to complete maneuver.	UPDATE PASTES
DATABL		Logical	Switch set true when bridge load data block exists.	CONTRO
DBUG		Logical	Run debug printout switch. Default false. If set true prints vehicle generation data, roadway to 6000 ft, buffers and tables, and bridge loading data.	INDATA REZONE SORPOS
DELHD	(2)		Increment value for each direction for use with headway, HDTAB, tables.	READ GEN
DELSD	(20)		Increment value for use with speed tables for each vehicle type.	READ GEN

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED PROGRAM	DIMENSION	UNITS/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
DELTIM		sec	Motion integration interval.	BRGLOD CONTRO INDATA ORDER PASTES READ UPDATE
DELWT	(20)		Increment value for use with weight tables.	GEN READ
DESGAP		ft	Desired gap between vehicles.	PASTES UPDATE
DISTLD	(50)		Load distribution histogram corresponding to load increments in table TAL.	INIT STAT
DISTTY	(20)		Vehicle type distribution histogram.	INIT STAT UPDATE
DPLTON	(10, 2)		Truck platoon distribution input for platoon generation.	GEN INDATA READ
DTL			Motion integration interval for output with bridge load data block.	ORDER
DXPOS	(50)	ft	Distance traveled by axle over the bridge during the integration interval.	ORDER UPDATE
DXPOSX	(50)	ft	Same as DXPOS ordered for output with bridge load data block.	ORDER
EXSPD		ft/sec	Excess speed permitted for passing.	INDATA

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNITS/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
EXSPD		ft/sec	Excess speed permitted for passing.	INDATA
FRTGAP		ft	Distance between maneuvering vehicle and the vehicle immediately forward.	PASPOS PASTES
FT	(4, 6)		Coefficients of the acceleration equation.	CALACC INDATA
GAP		ft	Distance between two vehicles.	PASPOS PASTES UPDATE
GAPFAC			Factor used to establish passing decision. GAPFAC $\frac{\text{VEHLEN(1)}}{\text{SDFAC}}$	BRGLOD PASTES UPDATE
GLAG		ft	Distance behind maneuvering vehicle and nearest vehicle in next lane.	PASPOS PASTES
GLEAD		ft	Distance ahead of maneuvering vehicle and nearest vehicle in the next lane.	PASPOS PASTES
HAFDEL		sec	One half of the integration interval.	BRGLOD UPDATE
HDFV		sec	Headway of the last vehicle generated in the forward direction.	CONTRO INIT
HDRV		sec	Headway of the last vehicle generated in the reverse direction.	CONTRO INIT
HDTAB	(40, 2)	sec	Headway distribution tables for vehicle generation.	GEN INDATA READ

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
IBAK	(400)		Buffer allocation table--forward traffic.	GEN GRAPH INIT PASPOS SORPOS UPDATE
IDPLTIN	(10)		Platoon distribution--sampled on the bridge.	CONTRO INIT STAT UPDATE
IEVENT			Bridge load event number.	INIT ORDER UPDATE
IEVNTX			Same as IEVENT for output with bridge load data block.	ORDER
IFWD			Buffer allocation table--forward traffic.	CONTRO GEN GRAPH INIT PASPOS SORPOS UPDATE
IGPLTON	(10, 2)		Platoon distribution at vehicle generation--forward and reverse.	GEN INIT STAT
INDX			Buffer allocation table--forward traffic.	CONTRO GEN GRAPH INIT PASPOS SORPOS UPDATE

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
IOUT			Output device number for bridge loading data.	INDATA ORDER
IPLTON	(2)		Platoon size counter.	GEN
ITV			Total number of vehicles generated.	GEN INIT STAT
ITY			Vehicle type-generated or or maneuvering vehicle.	GEN REZONE UPDATE
ITYPE	(400)		Vehicle type table.	GEN PASTES REZONE SORPOS UPDATE
IX			Random number function variable.	INDATA RANF
JBAK	(200)		Buffer allocation table-- reverse traffic.	GEN GRAPH INIT PASPOS SORPOS UPDATE
JFWD	(200)		Buffer allocation table--	GEN GRAPH INIT PASPOS SORPOS UPDATE
JHD			Headway table input variable maximum 2	READ
JNDX			Buffer allocation table-- reverse traffic.	GEN GRAPH INIT PASPOS SORPOS UPDATE

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
JOKE			Indicator for generation of forward (1), or reverse (2) vehicle.	CONTRO GEN
JPL			Read control for truck Platoon distribution table.	READ
JSD			Number of speed tables to be read in.	READ
JTY			Read control for vehicle classification table.	READ
JWT			Number of weight tables to be read.	READ
KLANE	(20)		Bridge load truck identification data--lane.	ORDER UPDATE
KSTAT	(400)		Vehicle passing status -1 follow 0 normal 1 passing	GEN PASPOS PASTES SORPOS UPDATE
KTYPE	(20)		Bridge load truck identification data--type.	ORDER UPDATE
LANE	(400)		Lane in which vehicle is traveling. Equal to 1-right, 2-left in forward direction, -2 right, -1 left in reverse direction.	CALACC GEN GRAPH PASPOS PASTES SORPOS UPDATE
LAST		Logical	Last data block indicator for each event.	CONTRO INIT ORDER
LHD	(2)		Number of headway table values, each direction.	GEN READ

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINE USING SYMBOL
LLANE	(20)		Same as KLANE for bridge load data output block.	ORDER
LNUM	(50)		Lane position of axle on the bridge--bridge load data.	ORDER UPDATE
LNUMX	(50)		Lane position of axle on the bridge--ordered for output with bridge load data.	ORDER
LSP			Number of speed table values.	GEN READ
LT			Number of load intervals.	BRGLOD INDATA
LTYPE	(20)		Bridge load truck identification data--truck type--output data block.	ORDER
LV			LT 1	BRGLOD READ UPDATE
LWT	(20)		Number of weight table values.	GEN READ
MD			Number of vehicle types Maximum 20, Default 11	BRGLOD INDATA READ
MPLTON			Platoon size counter.	CONTRO INIT UPDATE
MU			Index of maneuvering vehicle.	CALACC PASPOS PASTES REZONE UPDATE

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
MZ			Index and sign of Restricted Zone Grade	CALACC PASPOS REZONE UPDATE
ND			Number of directions of traffic, 1 forward traffic only, 2 bidirectional traffic. Default 1	BRGLOD CONTRO GRAPH INDATA PASPOS READ REZONE UPDATE
NL			Number of lanes of traffic.	INDATA REZONE UPDATE
NOAX	(20)		Number of axles per vehicle type.	INDATA UPDATE
NOAXL			Number of axles on bridge --bridge load data.	ORDER UPDATE
NOAXLX			Number of axles on bridge-- bridge load data output block.	ORDER
NR			Period counter.	CONTRO READ
NRAND			Input modifier to random number seed. Default 0	INDATA
NTH			Number of subperiods. Default 1	BRGLOD CONTRO INDATA
NTRUE			Number of trucks on bridge during an event.	INIT ORDER

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
NTRUKX			Same as NTRUR for bridge load data output block.	INIT ORDER
NUMD			Number of table values to be read.	
NZ			Number of restricted zones Max 5, Default 0	INDATA REZONE UPDATE
PLATON	(2)	Logical	Platoon generation indicator True if platoon is being generated.	GEN INIT
POS	(400)	ft	Position of vehicle.	CONTRO GEN GRAPH PASPOS PASTES REZONE UPDATE
POWER	(20)		Horsepower of vehicle.	CALACC INDATA
RDEND		ft	End of the roadway. For forward traffic only, equal to the end of the bridge, for bidirectional traffic, twice the position of the center of the bridge.	CONTRO GEN GRAPH PASPOS PASTES REZONE UPDATE
SAFDIS		ft	Minimum distance permitted between rear bumper of lead vehicle and front bumper of following vehicle.	INDATA PASTES
SDFAC			Factor used to calculate desired gap.	BRGLOD INDATA
SDTAB	(20, 20)	ft/sec	Speed table values for each type.	GEN INDATA READ

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
SPCK		ft	Space between lead and following vehicles.	
SPD	(400, 2)	ft/sec	Vehicle speed, current and generated.	CALACC GEN PASPOS PASTES SORPOS UPDATE
SPDLIM		ft/sec	Speed limit of road. This value is input as mi/hr and converted to ft/sec.	INDATA PASTES
SPDMAX		ft/sec	Maximum speed permitted.	INDATA PASPOS UPDATE
SPDMIN		ft/sec	Minimum speed permitted. This value is input as mi/hr and converted to ft/sec.	GEN INDATA UPDATE
SPDT	(20)	ft/sec	Speed of truck going on bridge--load data.	ORDER UPDATE
SPDX	(20)	ft/sec	Speed of truck going on bridge--load data output block.	ORDER
SUBPER		sec	Length of subperiod in seconds.	CONTRO READ
SUMHR		sec	Cumulative time of simulation.	CONTRO ORDER
SUMHRX		sec	Cumulative time of simulation for bridge load data output block.	ORDER
TAL	(51)	lb	Load increment values.	BRGLOD STAT

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
TALINC		lb	Load increment for which load histogram is made.	BRGLOD INDATA UPDATE
TIMET	(20)	sec	Time of truck going on bridge.	ORDER UPDATE
TIMEX	(20)	sec	Time of truck going on bridge--for load data output block.	ORDER
TIMLIM		sec	Total time to be simulated.	BRGLOD CONTRO INDATA
TOTIM		sec	Total simulated time.	CONTRO INIT STAT
TRKLIM		ft/sec	Speed limit for trucks. This value is input as mi/hr and is converted to ft/sec.	INDATA PASTES
V	(5)	ft	Beginning of restricted zone, upgrade in forward direction.	INDATA REZONE
VEHLEN	(20)	ft	Vehicle length by type.	BRGLOD GEN INDATA PASPOS PASTES REZONE UPDATE
W	(5)	ft	End of restricted zone, upgrade in forward direction.	INDATA REZONE
WEIT	(50)	lb	Axle weight--bridge load data.	ORDER UPDATE

TABLE 1. LIST OF VARIABLES (CONTINUED)

SYMBOL USED IN PROGRAM	DIMENSION	UNIT/ TYPE	DESCRIPTION	SUBROUTINES USING SYMBOL
WEITX	(50)	lb	Axle weight bridge load data ordered for output data block.	ORDER
WGT	(400)	lb	Vehicle weight.	CALACC GEN UPDATE
WGTT	(20)	lb	Bridge load truck identification data--weight.	ORDER UPDATE
WGTX	(20)	lb	Bridge load truck identification data--weight--output data block.	ORDER
WTAB	(30, 20)	lb	Weight table.	GEN INDATA READ
X	(5)	ft	Beginning of restricted zone downgrade in forward direction.	INDATA REZONE
XPOS	(50)	ft	Position of axle loads bridge load data.	ORDER UPDATE
XPOSX	(50)	ft	Position of axle loads ordered for output data block.	ORDER
Y	(5)	ft	End of restricted zone downgrade in forward direction.	INDATA REZONE
Z	(5)	ft	Percent grade of restricted zone. 0 for curve	CALACC INDATA PASPOS REZONE

Program Routines

The following describes the Bridge Load Generator in terms of its individual subroutines. The program functions, equations solved, program interfaces are described, followed by the program flow chart and program listing. Figure 10 Program Interfaces depicts the hirearchy of subroutine calls. Figure 11 depicts the designating notation for relative positional relationships between vehicles.

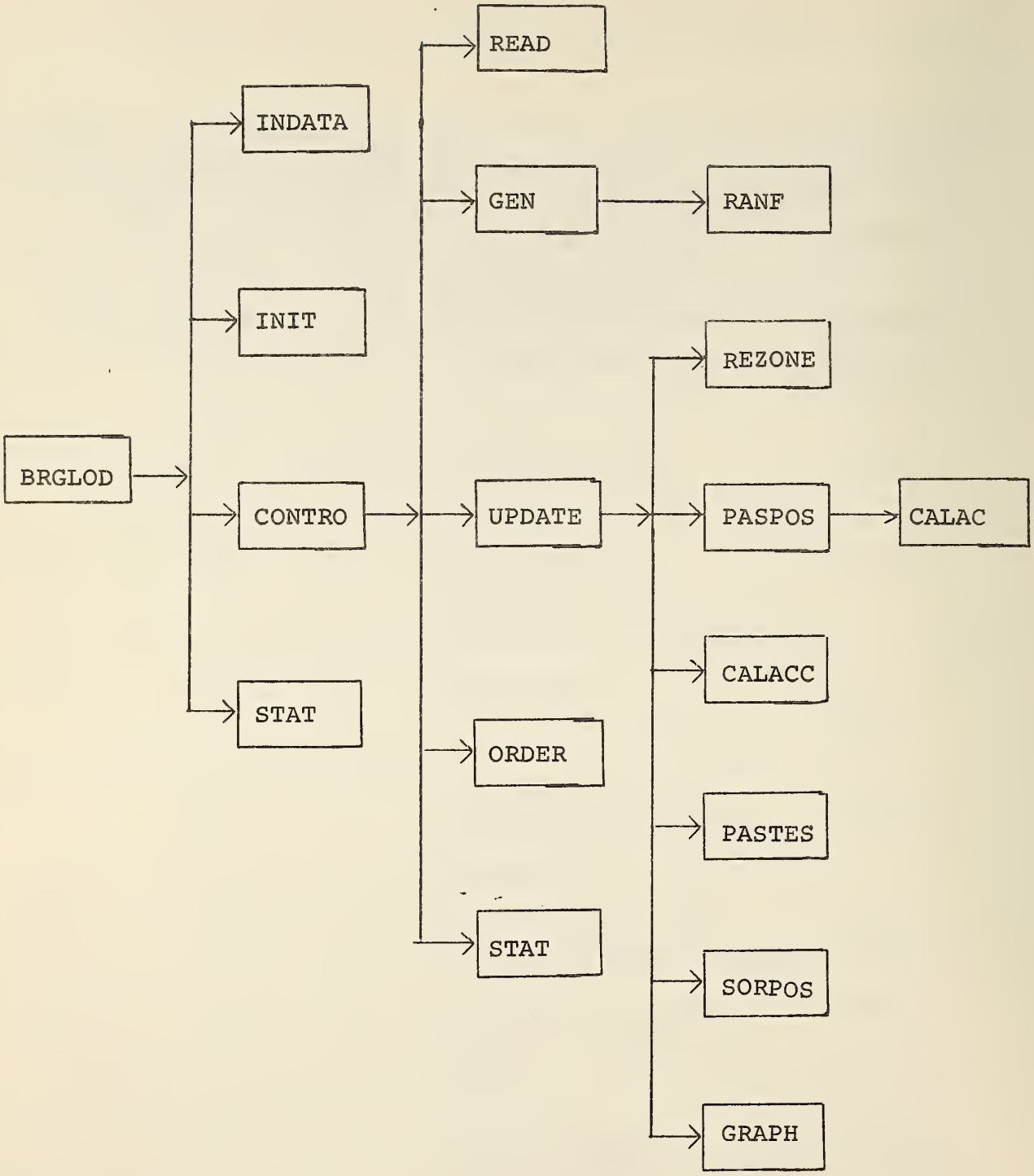
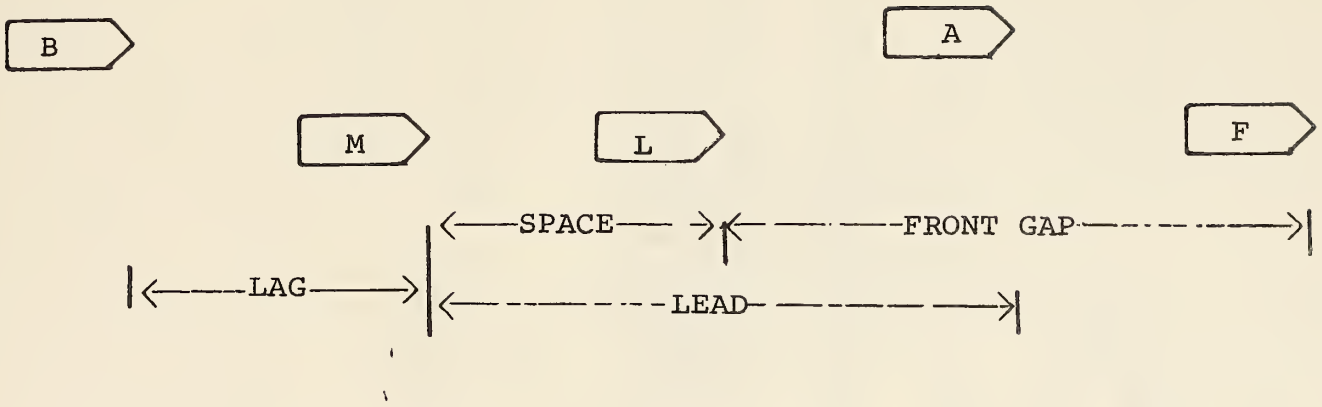


Figure 10. Program Interfaces



Lag = Distance between maneuvering vehicle and vehicle behind it in the next lane.

Lead = distance between maneuvering vehicle and vehicle ahead in the next lane

Space = distance between maneuvering vehicle and vehicle ahead in same lane

Front Gap = distance between vehicle ahead of maneuvering vehicle and the vehicle ahead of it, all in the same lane

Figure 11. Maneuvering Vehicles

The program main routine, BRGLOD, reads in all simulation data through the INDATA subroutine, calculates:

$$\text{HAFDEL} = \frac{1}{2}\ \text{DELTIME}$$

$$\text{GAPFAC} = \frac{\text{Car Length}}{\text{SDFAC}}$$

BREND = BRPOS + BRLEN

BRST = BRPOS - BRLEN

RDEND = BREND for forward traffic only

RDEND = 2 x BRPOS for bidirectional traffic

The load interval table is calculated and simulation heading is printed. Data and tables are initialized, simulation run, and the final statistics are printed out. Subroutines called are:

INDATA

INIT

CONTRO

STAT

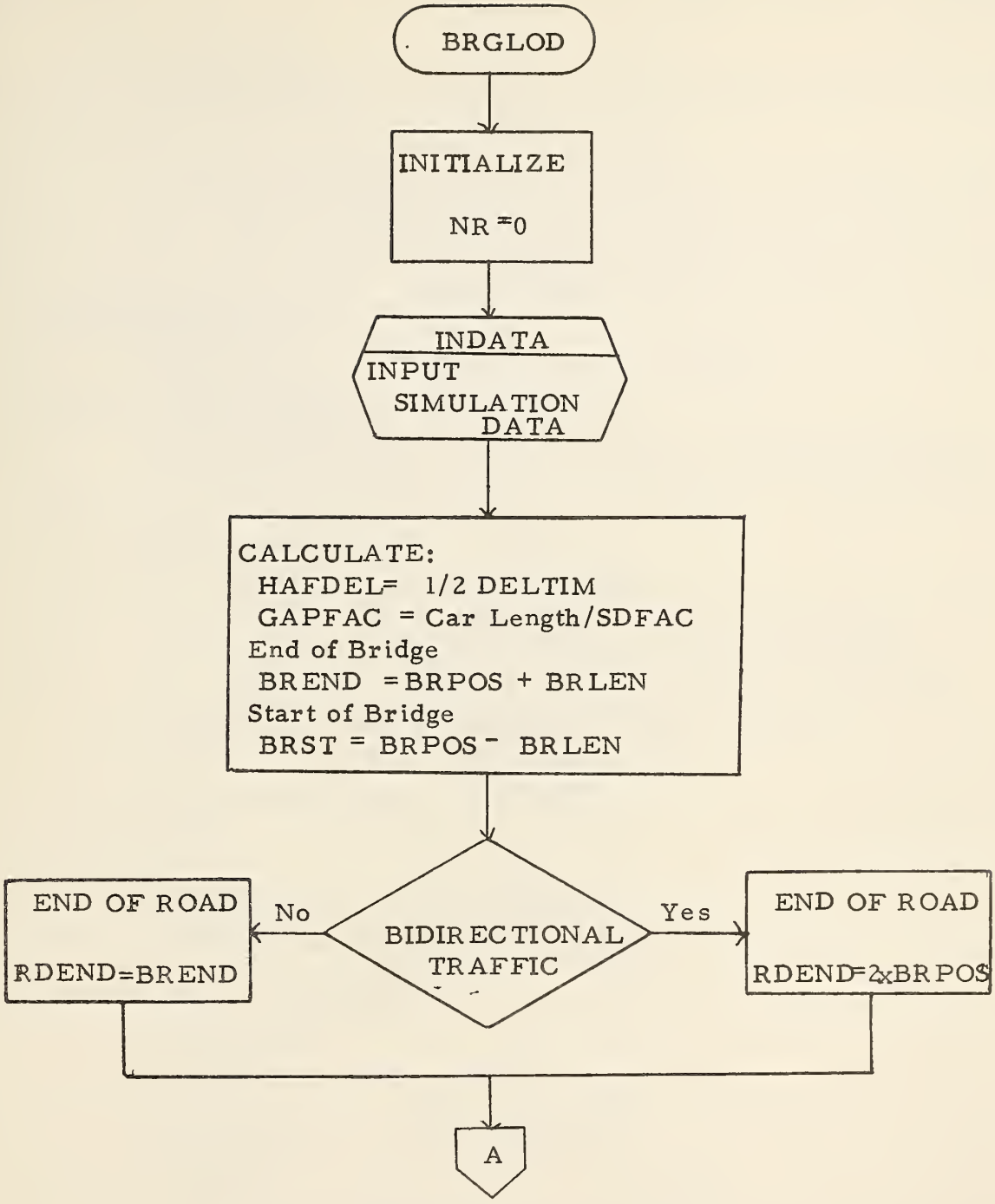


Figure 12. BRGLOD Program
Flow Chart

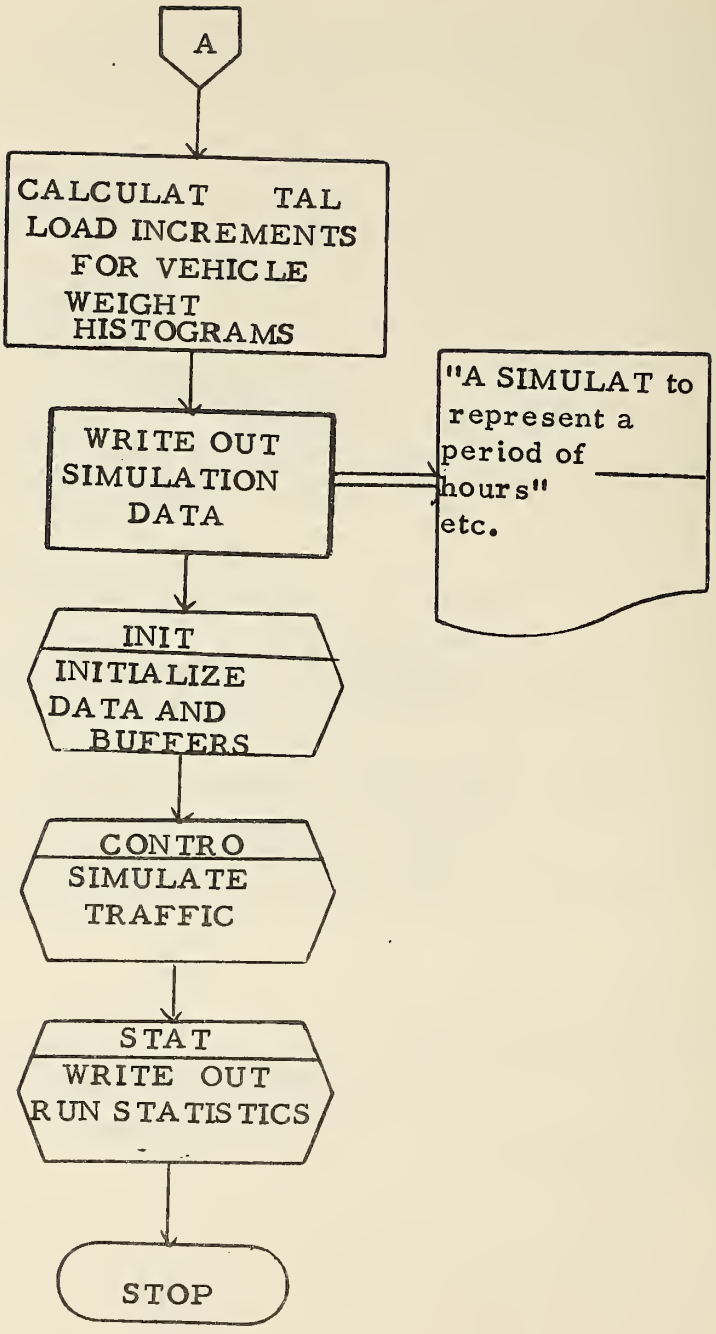


Figure 12. BRGLOD Program Flow Chart (continued)

```
      C
      C   VEHICLE DATA
001           COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
             1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
      C
      C   BRIDGE, ROAD AND TIME DATA
002           COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
             1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
             2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
             3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
             4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
      C
      C   STATISTICAL DATA
0003          COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
             1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
      C
      C   VEHICLE GENERATION DISTRIBUTION DATA
0004          COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
             1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
             2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
             3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
      C
      C   BRIDGE LOADING DATA
0005          COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
             1     XPOS(50), DXPOS(50), ACCLR(50),
             2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
006           COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
             1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
             2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
0007          DIMENSION JFWD(200),JBAK(200),JNDX(200)
008           EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
             1           (JNDX(1),INDX(201))
      C
0009          DIMENSION REF20(20)
      C
0010          NR=0
      C
0011          CALL INDATA
012           HAFDEL=.5*DELTIM
0013          GAPFAC=VEHLEN(1)/SDFAC
      C
0014          BREND   =BRPOS+BRLEN
015           BRST   =BRPOS-BRLEN
      C
016           RDEND=BREND
0017          IF(ND.EQ.2)  RDEND = 2.*BRPOS
      C
0018          LV=LT+1
019           DO 10 M=1,LV
0020          TAL(M)=(M-1) *TALINC
021       10  CONTINUE
      C
022           TIMLM = TIMLIM/3600.
0023          WRITE(6,120) TIMLM, BRPOS,TALINC,TAL(LT),NTH,MD
```

```
0024          120 FORMAT (1H1,// 15X,'A SIMULATION TO REPRESENT A PERIOD OF ',F8.1, 0C
                 $' HOURS.'//5X,
                 $'VEHICLES ARE GENERATED ',F6.0,' FEET FROM BRIDGE-CENTER.  WEIGHTS0C
                 $ ON BRIDGE'/,5X'ARE SUMMED AND COUNTED FOR LOAD INCREMENTS OF',F6.0C
                 $0,' POUNDS UP TO',F7.0,'.'/,15X,I2,' PERIOD TYPES AND ',I2,       0C
                 $' VEHICLES TYPES ARE CONSIDERED.')                                0C
              C
0025               CALL INIT
              C
0026               CALL CONTRO
0027               CALL STAT
              C
0028           1000 STOP
0029               END
```

```
001               BLOCK DATA
          C    VEHICLE GENERATION DISTRIBUTION DATA
002               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                 3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
003               COMMON /RANDOM/ IX,IY,YFL                                        00
          C
004               DIMENSION A(20),B(24),C(100),D(20),E(100),A1(40),B1(80),
                 1C1(400),D1(600),E1(20),
                 2D2(30),D3(30),D4(30),D5(30),D6(30),D7(30),D9(330),C2(260)
005               EQUIVALENCE (NOAX(1),A(1)),(FT(1),B(1)),(AXWT(1),C(1)),
                 1(VEHLEN(1),D(1)),(AXPOS(1),E(1)),(AFS(1),A1(1)),(HDTAB(1),B1(1)),
                 2(SDTAB(1),C1(1)),(WTAB(1),D1(1)),(DPLTON(1),E1(1)),(D2(1),D1(91))
                 3,(C2(1),C1(141)),(D3(1),D1(121)),(D4(1),D1(151)),
                 4(D5(1),D1(181)),(D6(1),D1(211)),(D7(1),D1(241)),
                 5(D9(1),D1(271))
006               DATA SDFAC,SAFDIS,SUBPER/15.0,10.0,60.0/
007               DATA IX/246801357/                                               00
          C
008               DATA DELHD/.05,.05/,LHD/21,21/,LSP/11*11/
009               DATA LWT/2,21,22,8*21,9*0/
010               DATA DELSD/11*0.1,9*0.0/
011               DATA DELWT/1.0,19*.05/
012               DATA NOAX/3*2,8*3,9*0/
013               DATA VEHLEN/19.,23.,28.,38.,46.,54.,46.,50.,54.,46.,54.,9*0.0/
          C
          C      E1=DPLTON
          C
014               DATA E1/1.0,9*0.0,1.0,9*0.0/
          C
          C      A1=AFS OR AFR
          C
015               DATA A1/.83,.8752,.8852,.8882,.8912,.8942,.9064,.9186,.9532,.9878,
                 110*1.0,.83,.8752,.8852,.8882,.8912,.8942,.9064,.9186,.9532,.9878,
                 210*1.0/
          C
          C      B=FT
          C
016               DATA B/14.7,.10,0.0,140.,11.7,.09,0.0,120.,13.0,.247,.00118,90.,
                 19.3,.198,.00107,44.,5.7,.15,.001,28.,4.,.102,.00065,38./
          C
          C      C=AXWT
          C
017               DATA C/2*.5,3*0.0,.25,.75,3*0.0,.25,.75,3*0.0,.2,.4,.4,2*0.0,
                 1.2,.4,.4,2*0.0,
                 1.2,.5,.3,2*0.0,.2,.4,.4,2*0.0,.1,.4,.5,2*0.0,.1,.3,.6,2*0.0,
                 2.2,.4,.4,2*0.0,.2,.4,.4,2*0.0,45*0.0/
          C
          C      E=AXPOS
          C
018               DATA E/ 3.0,14.0,3*0.0,      4.0,19.0,3*0.0,      4.0,20.0,3*0.0,
                 1         4.,15.5,32.,2*0.,    4.,15.5,42.,2*0.,    4.,15.5,48.,2*0.,
                 2         4.,15.5,38.,2*0.,    4.,15.5,42.,2*0.,    4.,15.5,46.,2*0.,
```

```
     3       4.,17.5,38.0,2*0.,  4.0,17.5,46.,2*0.,  45*0.0/
C
C     B1=HDTAB
C
      DATA B1/.4,.5,.6,.7,.8,.9,1.,1.1,1.3,1.5,1.6,1.8,2.,2.1,2.5,
     12.8,3.,3.5,4.1,5.2,5.6,19*0.0,.4,.5,.6,.7,.8,.9,1.,1.1,1.3,1.5,1.6
     2,1.8,2.,2.1,2.5,2.8,3.,3.5,4.1,5.2,5.6,19*0.0/
C
C     C1=SDTAB(1-7,20)
C
      DATA C1/40.,66.2,71.,74.3,77.1,80.,82.7,85.5,89.,93.8,120., 9*0.0,
     140.,62.,66.,68.8,71.2,73.5,75.8,78.1,81.,85.,106.6, 9*0.0,
     230.,55.6,60.2,63.8,66.6,69.5,72.,74.9,78.1,83.,109., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0/
C
C     C2=SDTAB(8-20,20)
C
      DATA C2/31.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120.,
     1 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     331.6,60.3,65.7,69.5,72.7,75.5,78.6,82.,85.5,91.,120., 9*0.0,
     4180*0.0/
C
C     D1=WTAB(1-3,30)
C
      DATA D1/2*3000.,28*0.0,3000.,6500.,7300.,8000.,8700.,9800.,10000.,
     110400.,11000.,11800.,12500.,13400.,14500.,16000.,17800.,18800.,
     219800.,20800.,21300.,22600.,25000., 9*0.0,
     3  12000., 16000., 17600., 19200., 20900., 22600., 25050., 27500.,
     4  29750., 32000., 33250., 34500., 35300., 36100., 37050., 38000.,
     5  39550., 41100., 46650., 52200., 55150., 58100., 8*0./
C
C     D2=WTAB(4,30)
C
      DATA D2/12000.,16900.,18500.,19400.,20000.,20200.,21500.,22400.,
     723400.,24800.,26800.,29300.,30800.,31600.,32300.,33300.,34100.,
     835300.,37000.,39600.,59700., 9*0.0/
C
C     D3=WTAB(5,30)
C
      DATA D3/12000.,16900.,18500.,19400.,20000.,20200.,21500.,22400.,
     723400.,24800.,26800.,29300.,30800.,31600.,32300.,33300.,34100.,
     835300.,37000.,39600.,59700., 9*0.0/
C
C     D4=WTAB(6,30)
C
      DATA D4/12000.,16900.,18500.,19400.,20000.,20200.,21500.,22400.,
     723400.,24800.,26800.,29300.,30800.,31600.,32300.,33300.,34100.,
     835300.,37000.,39600.,59700., 9*0.0/
C
```

```
      C     D5=WTAB(7,30)
      C
026            DATA D5/15000.,19800.,21300.,22700.,
              824000.,25800.,28500.,32000.,34400.,36700.,38900.,41700.,44600.,
              947100.,48800.,52600.,55000.,57000.,58800.,61000.,65900., 9*0.0/
      C
      C     D6=WTAB(8,30)
      C
027            DATA D6/15000.,19800.,21300.,22700.,
              824000.,25800.,28500.,32000.,34400.,36700.,38900.,41700.,44600.,
              947100.,48800.,52600.,55000.,57000.,58800.,61000.,65900., 9*0.0/
      C
      C     D7=WTAB(9,30)
      C
028            DATA D7/15000.,19800.,21300.,22700.,
              824000.,25800.,28500.,32000.,34400.,36700.,38900.,41700.,44600.,
              947100.,48800.,52600.,55000.,57000.,58800.,61000.,65900., 9*0.0/
      C
      C     D9=WTAB(10-20,30)
      C
029            DATA D9/21000.,23000.,25000.,26000.,27000.,29500.,34700.,45200.
              1,52000.,56800.,60000.,62700.,65000.,66300.,67300.,68200.,69700.,
              270200.,71000.,71500.,83900., 9*0.0,
              821000.,23000.,25000.,26000.,27000.,29500.,34700.,45200.
              1,52000.,56800.,60000.,62700.,65000.,66300.,67300.,68200.,69700.,
              270200.,71000.,71500.,83900., 9*0.0,270*0.0/
      C
030            DATA POWER/100.,136.,157.,3*165.,2*172.,3*184.,9*0.0/
031            END
```

CALACC

The subroutine CALACC calculates the acceleration according to an equation suggested by Kobett of M.R.I. which is suitable for vehicles on level and small grades. The equation

$$a = C_o + C_1 v + C_2 v^2 + C_3 \tan\theta \qquad (1)$$

can be solved by using a table of values for the coefficients C (table FT) which depend on the weight to horsepower ratio. The use of the $\tan\theta$ is useful because it is simply .01 multiplied by the value of the grade as understood by highway engineers. The acceleration is not permitted to exceed the value ACCEL which is read in at the beginning of the simulation. For reverse direction traffic, acceleration is set negative. Acceleration is stored in the array ACC. No subroutines are called by this program.

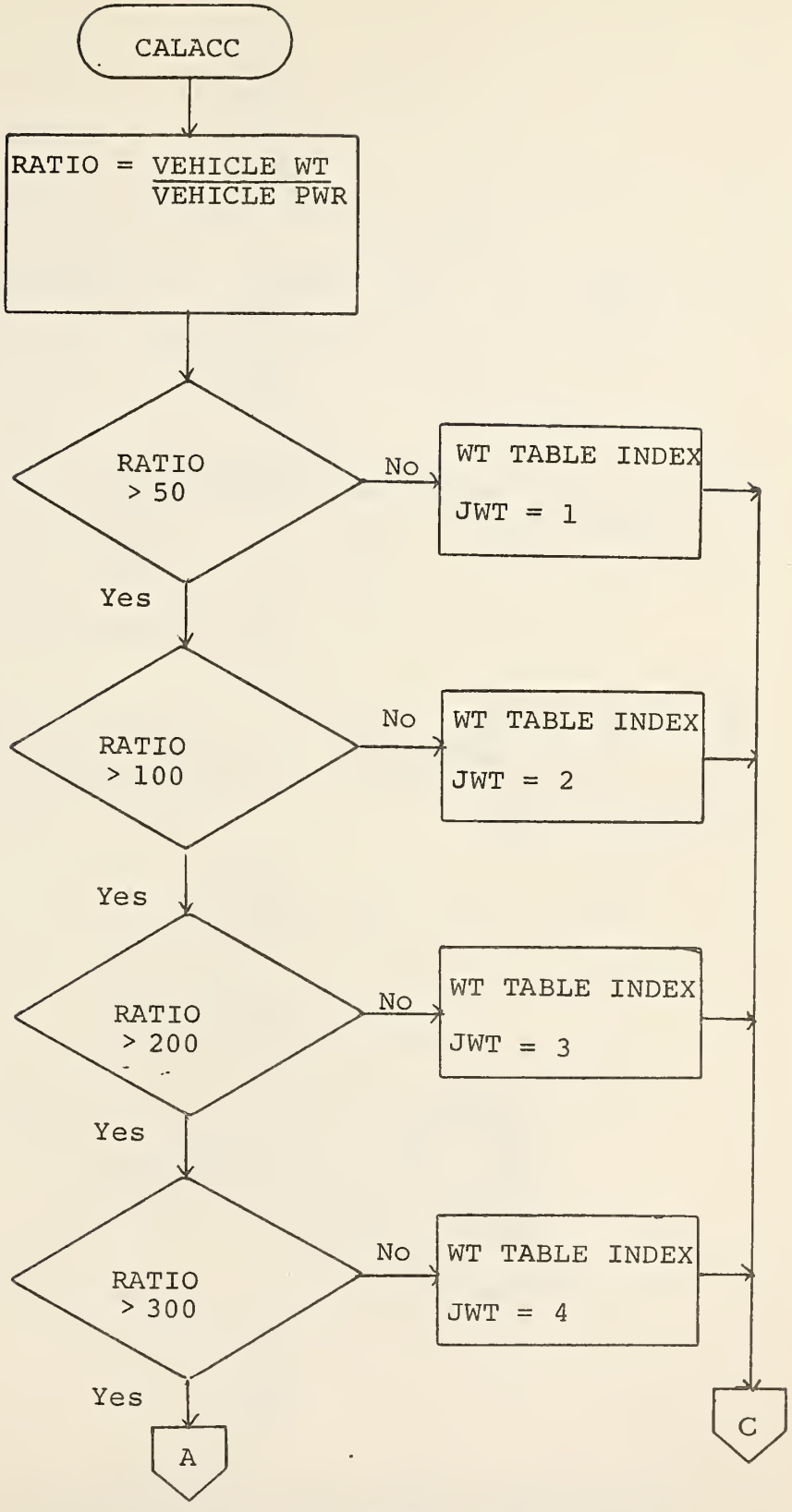


Figure 13. CALACC Program Flow Chart

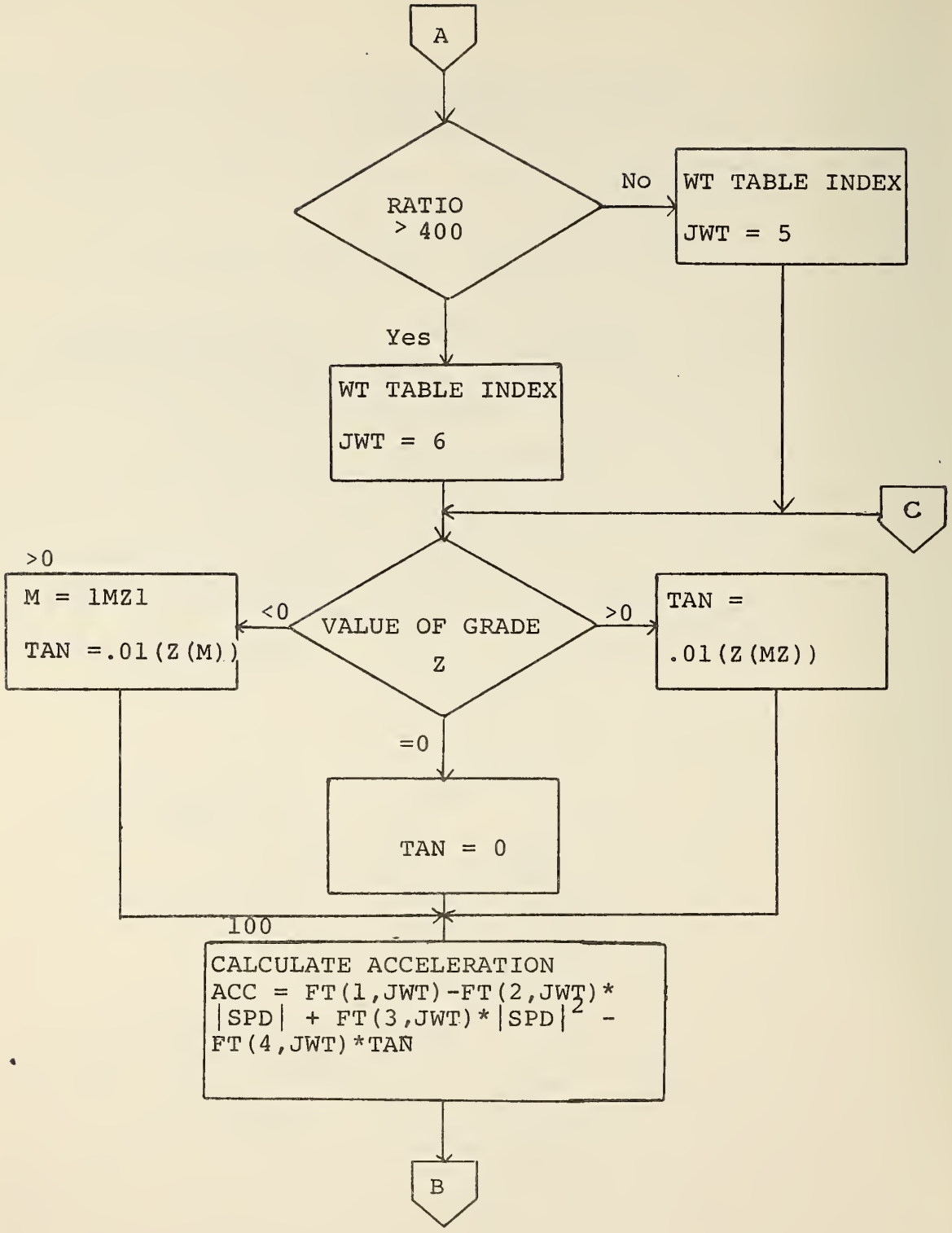


Figure 13. CALACC Program Flow Chart (Continued)

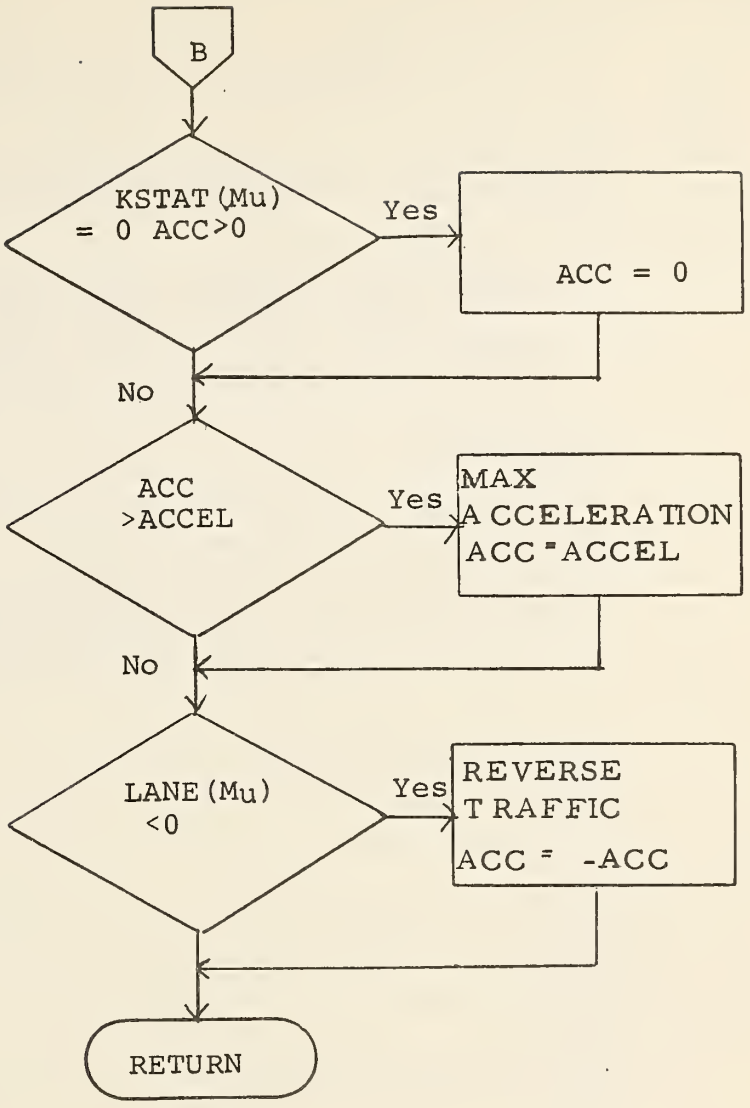


Figure 13. CALACC Program Flow Chart (Continued)

```
001               SUBROUTINE CALACC                                          00
                  C
                  C     CALCULATES ACCELERATION FOR PASSING VEHICLE.         00
                  C
                  C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                  C
                  C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                  1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                  2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                  3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                  4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                  C
                  C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                  1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                  C
                  C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                  3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                  C
                  C
006               RATIO=WGT(MU)/POWER(ITY)                                   00
007               IF(RATIO.GT.50) GO TO 10                                   00
008               JWT=1                                                      00
009               GO TO 60                                                   00
010              10 IF(RATIO.GT.100) GO TO 20                                00
011               JWT=2                                                      00
012               GO TO 60                                                   00
013              20 IF(RATIO.GT.200) GO TO 30                                00
014               JWT=3                                                      00
015               GO TO 60                                                   00
016              30 IF(RATIO.GT.300) GO TO 40                                00
017               JWT=4                                                      00
018               GO TO 60                                                   00
019              40 IF(RATIO.GT.400) GO TO 50                                00
020               JWT=5                                                      00
021               GO TO 60                                                   00
022              50 JWT=6                                                    00
023              60 IF(MZ)70,80,90                                           00
024              70 M=IABS(MZ)                                               00
025               TAN=-.01* Z(M)
026               GO TO 100                                                  00
027              80 TAN=0.0                                                  00
028               GO TO 100                                                  00
029              90 TAN=.01* Z(MZ)
030             100 VALSPD=ABS(SPD(MU,1))                                    00
031               SPDSQ=VALSPD*VALSPD                                        00
032               ACL=FT(1,JWT)-FT(2,JWT)*VALSPD+FT(3,JWT)*SPDSQ-FT(4,JWT)*TAN      0
033               IF(ACL.GT.ACCEL) ACL=ACCEL
```

```
034              IF(LANE(MU).LT.0) ACL=-ACL
035              ACC(MU) = ACL
036              RETURN                                                  00
037              END                                                     00
```

CONTRO

This subroutine controls the generation of new vehicles, reading in of subperiod data, vehicle motion integration, the writing out of bridge loading data, and the hourly printing of simulation statistics through calls to the following subroutines:

READ
GEN
UPDATE
ORDER
STAT

Simulation time is started when the first vehicle enters the bridge and the following is printed out: "SIMULATION START AT (current time) SECONDS, END AT START + (TIMLIM) SECONDS. At the end of the simulation run the statistical data for the last platoon is stored.

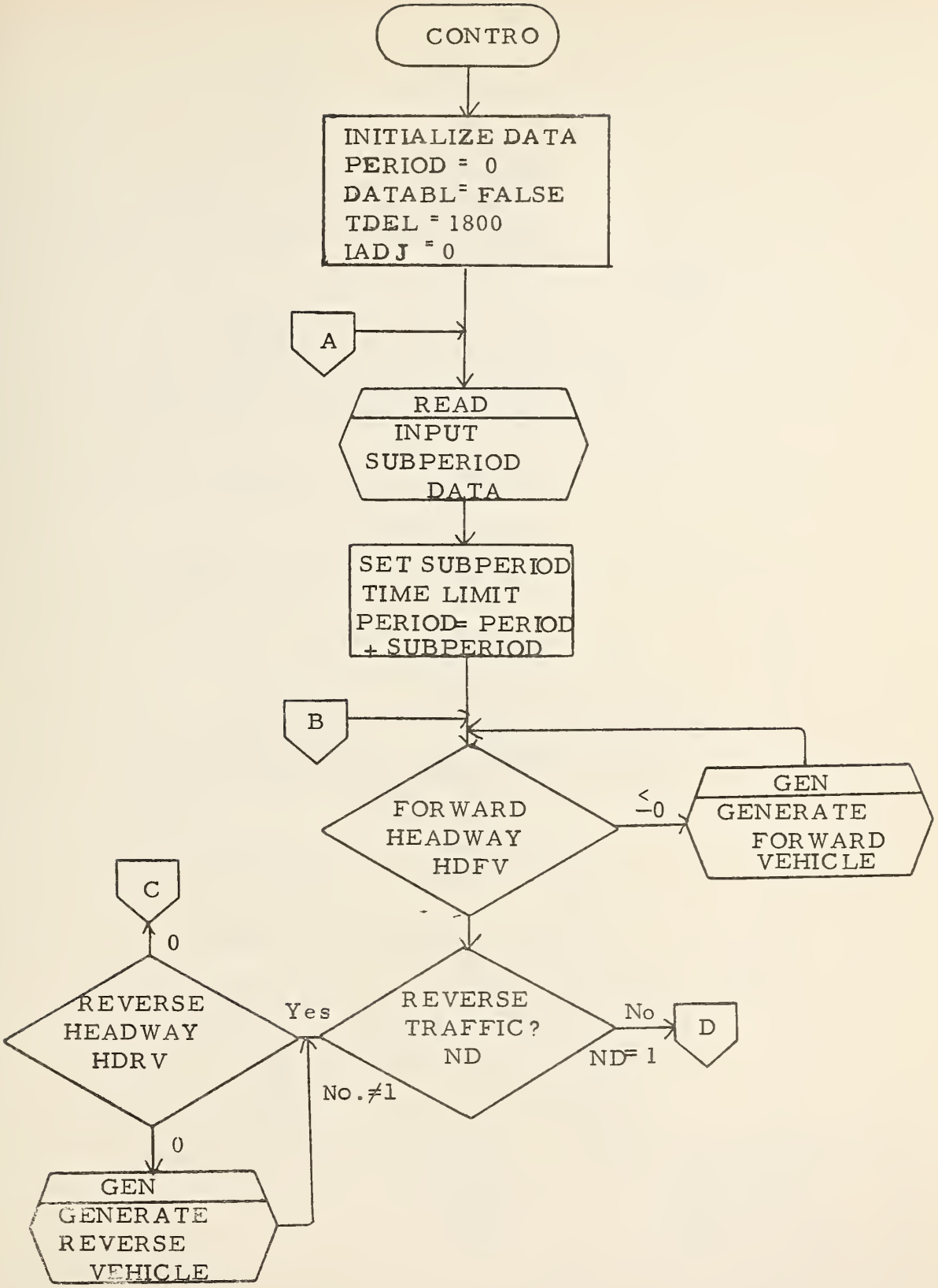


Figure 14. CONTRO Program Flow Chart

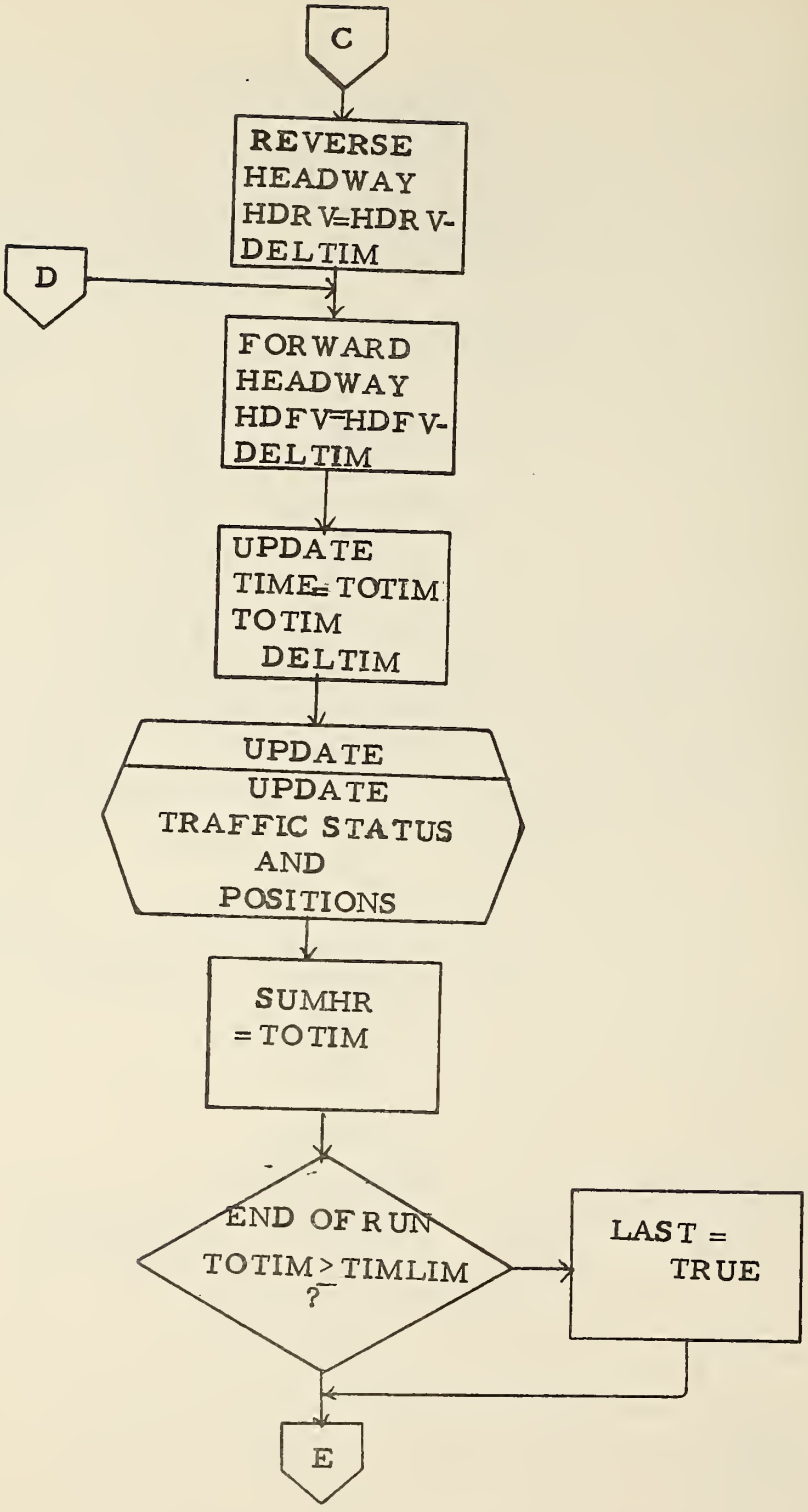


Figure 14. CONTRO Program Flow Chart (Continued)

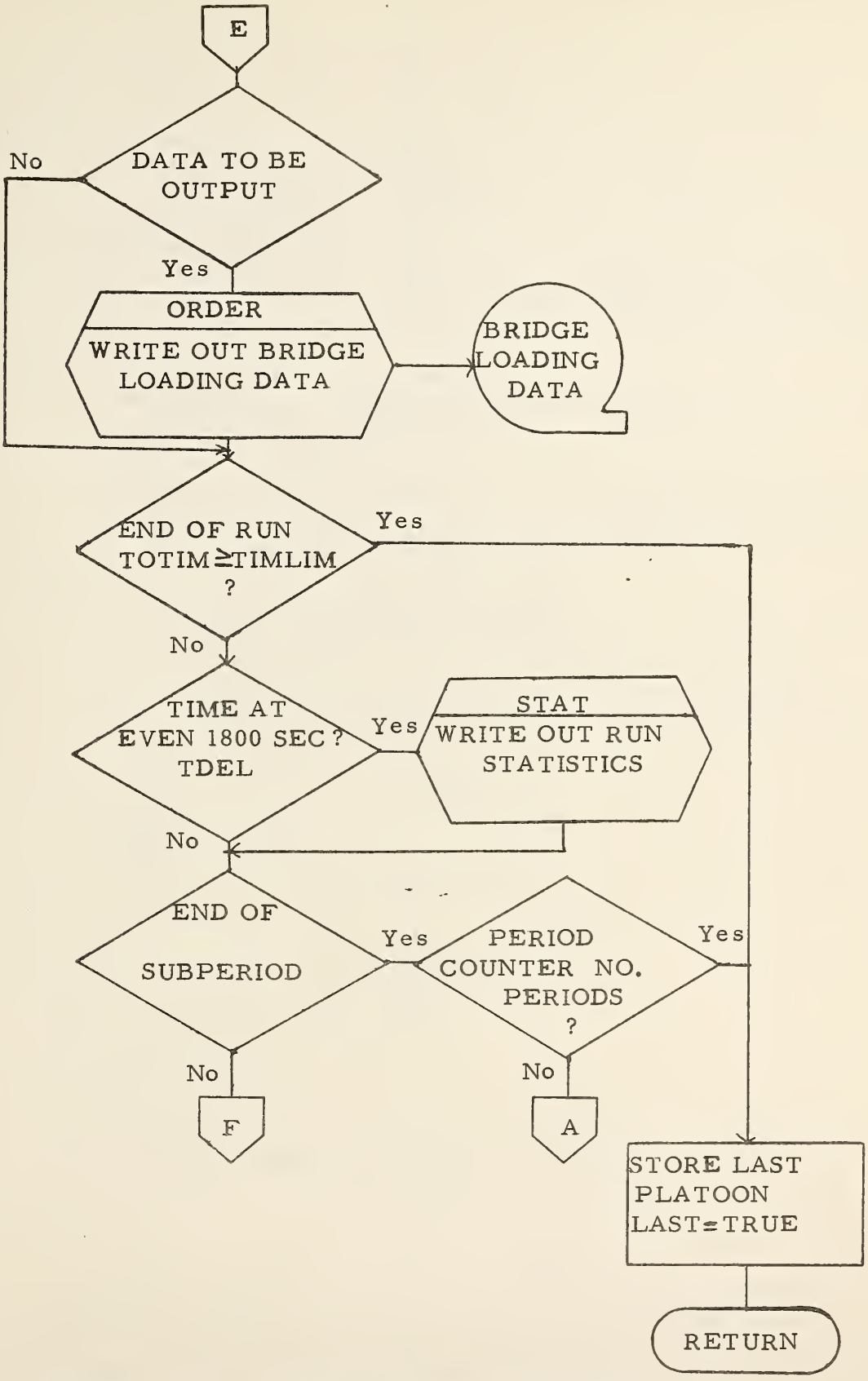


Figure 14. CONTRO Program Flow Chart (Continued)

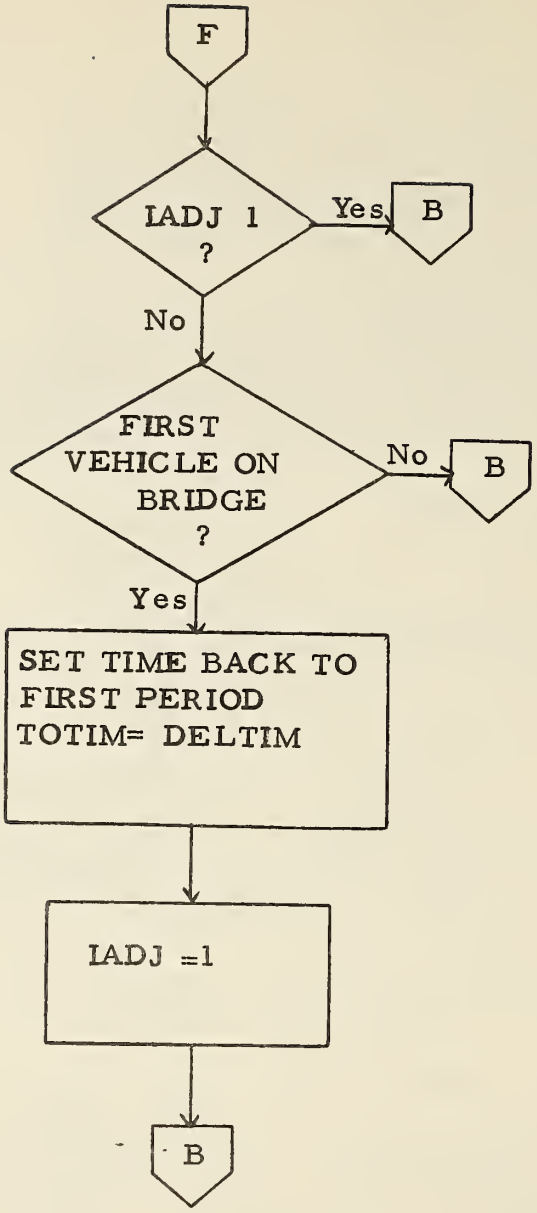


Figure 14. CONTRO Program Flow Chart (Continued)

```
      C
001           SUBROUTINE CONTRO                                          00
      C
      C     CONTROLS CALLING OF SUBROUTINES, INITIATES PARAMETERS.       00
      C
      C   VEHICLE DATA
002           COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
     1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
      C
      C   BRIDGE, ROAD AND TIME DATA
003           COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
     1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
     2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
     3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
     4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
      C
      C   STATISTICAL DATA
004           COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
     1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
      C
      C   VEHICLE GENERATION DISTRIBUTION DATA
005           COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
     1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
     2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
     3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
      C
      C   BRIDGE LOADING DATA
006           COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
     1     XPOS(50), DXPOS(50), ACCLR(50),
     2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
007           COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
     1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
     2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
      C
008           LOGICAL DATABL, FIRST, LAST, DBUG
      C
009           PERIOD = 0.0
010           DATABL = .FALSE.
011           TDEL = 1800.0
012           IADJ = 0
      C
013           GO TO 32
      C
      C
      C   PERIOD COUNTER = NO. OF PERIODS ?
      C
014     30    IF (NR.EQ.NTH) GO TO  20                                   00
      C
015           START=TOTIM                                                00
      C
      C   READ IN PERIOD DATA
      C
016        32 CONTINUE
017           CALL READ                                                  00
```

```
018               PERIOD = PERIOD + SUBPER
          C
019              10 CONTINUE
020             140 IF (HDFV.GT.0)  GOTO 120
021                 JOKE=1                                                01
          C
          C    GENERATE FORWARD VEHICLE
          C
022                 CALL GEN                                              01
023                 GO TO 140
          C
024             120 IF (ND.EQ.1)  GOTO 60
025             130 IF (HDRV.GT.0)  GO TO 50
026                 JOKE=2                                                01
          C
          C    GENERATE REVERSE VEHICLE
          C
027                 CALL GEN                                              01
028                 GO TO 130
029              50 HDRV=HDRV-DELTIM                                      01
030              60 HDFV=HDFV-DELTIM                                      01
031                 TOTIM=TOTIM+DELTIM                                    01
032                70 CALL UPDATE(DATABL)
033                 SUMHR = TOTIM
034                 IF (TOTIM.GE.TIMLIM)  LAST = .TRUE.
035                 IF (DATABL)  CALL ORDER
036                 IF (TOTIM.GE.TIMLIM)  GO TO 20
          C
037                 ITIMX = TOTIM/TDEL
038                 TIMY = TOTIM - ITIMX*TDEL
039                 IF(TIMY.LT.1.0)  CALL STAT
040                 IF (TOTIM.GE.PERIOD)  GO TO 30
          C
041                 IF(IADJ.EQ.1)  GO TO 10
042                 IFD = IFWD(1)
043                 M = INDX(IFD)
044                 IF (POS(M).GT.BRST)  GO TO 75
045                 GO TO 10
046                75 CONTINUE
047                 WRITE (6,1075)  TOTIM, TIMLIM
048              1075 FORMAT('OSIMULATION START AT',F10.1,' SECONDS, END AT START +',
                    1     F10.1,' SECONDS')
049                 TOTIM = DELTIM
050                 IADJ = 1
051                 GO TO 10
          C
052                20 CONTINUE
          C
          C    STORE LAST PLATOON
          C
053                 IF(MPLTON.GT.10)  MPLTON = 10
054                 IDPLTN(MPLTON) = IDPLTN(MPLTON) + 1
          C
055                 LAST = .TRUE.
```

```
056                 RETURN
057                 END                                                    00
```

GEN

This routine calculates all the data associated with the generation of vehicles. Vehicles are generated by using a random number with each of the following tables:

AFR = Vehicle type distribution by direction

DPLTON = Platoon size distribution by direction

HDTAB = Headway distribution by direction

SDTAB = Speed distribution by type

WTAB = Weight distribution by type

These tables determine the characteristics of each individual vehicle when it is initiated into the simulation. Generation of the next vehicle is initiated when the headway for the preceeding vehicle has elapsed, that is, the headway associated with each vehicle is considered to be behind it. The entering vehicle is assumed to be in the right-hand lane when it starts on the roadway and in a free operation state (neither following nor passing). The vehicle is placed on the roadway at the distance it would have traveled from when the headway = 0, that is,

Position = - Headway x Speed

If there is no room in the vehicle data tables for an additional entry, the GEN simply returns without generating a vehicle. Because the headway remains negative, this routine is called every Δt until space is available. The new headway is then calculated as the sum of the old (0 or negative) and the new value.

Subroutines called by GEN are:

RANF

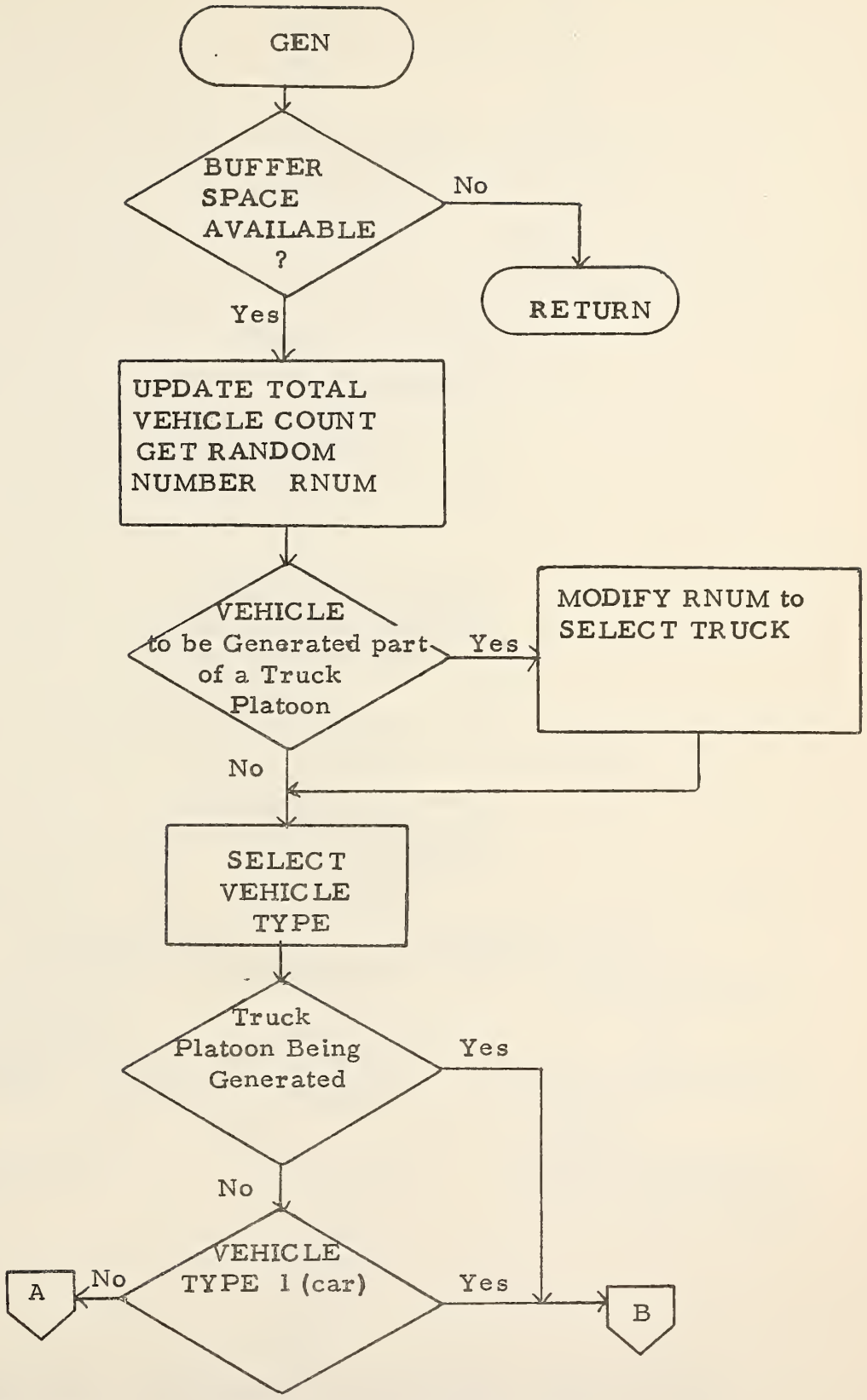


Figure 15. GEN Program Flow Chart

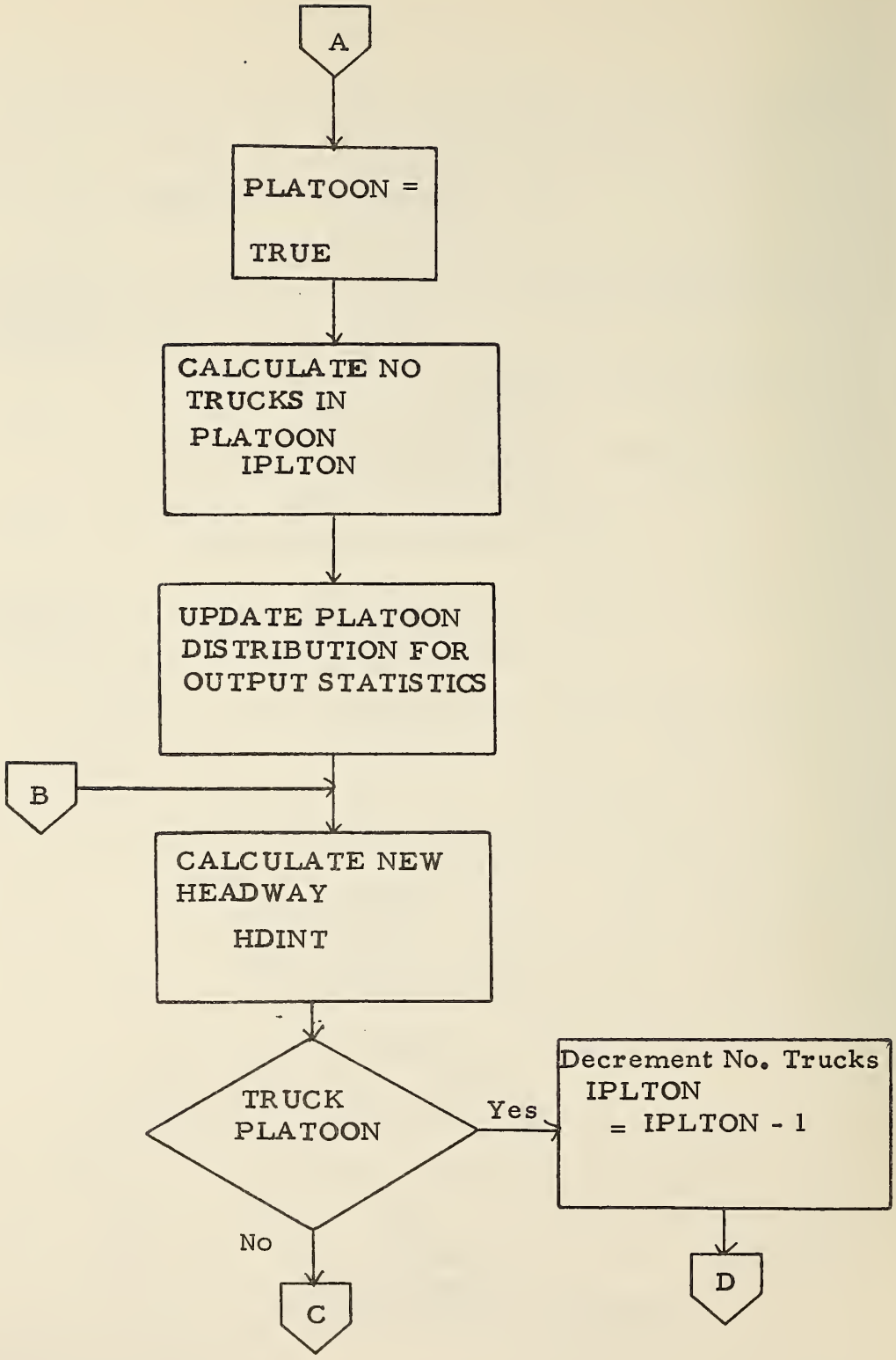


Figure 15. GEN Program Flow Chart (Continued)

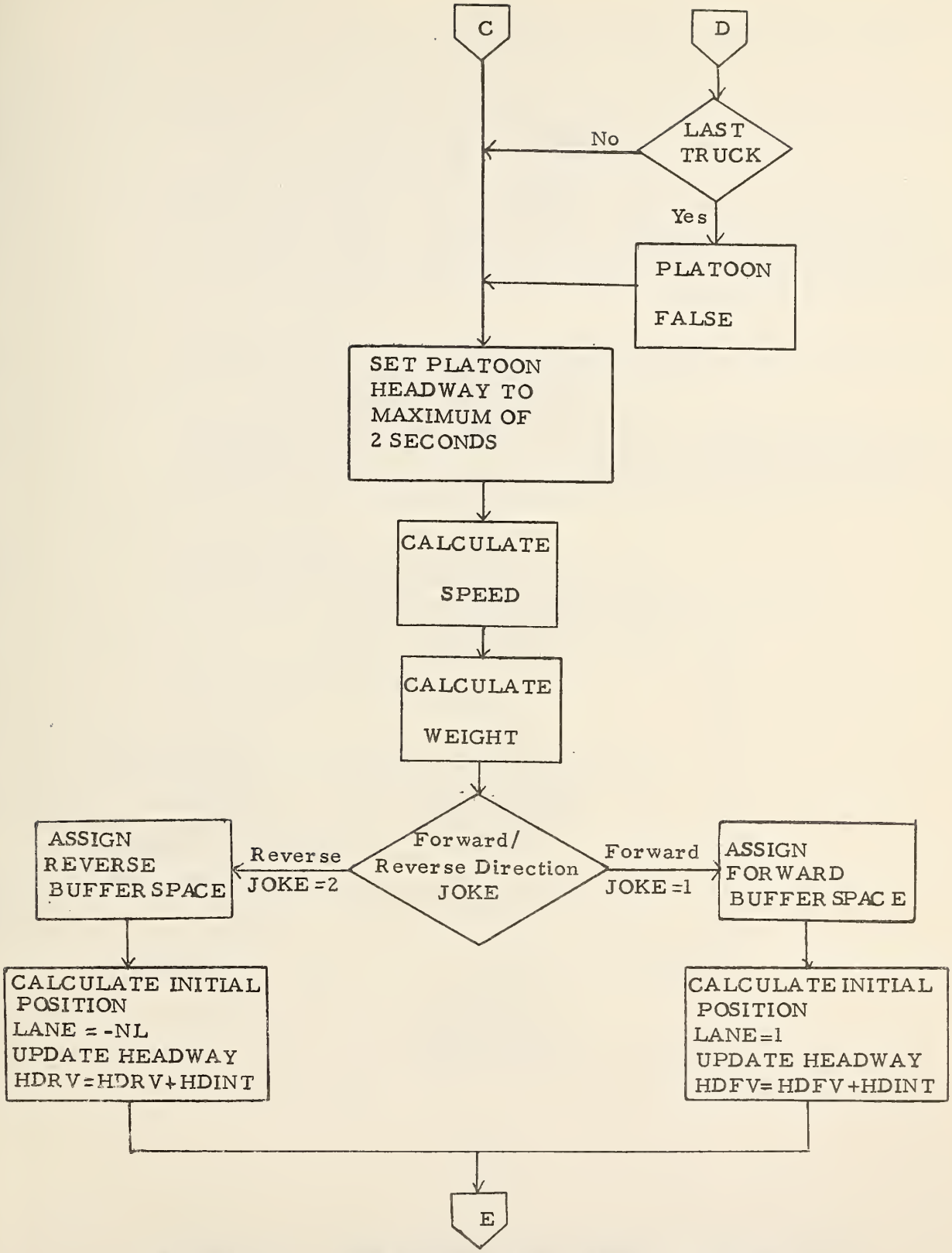


Figure 15. GEN Program Flow Chart (Continued)

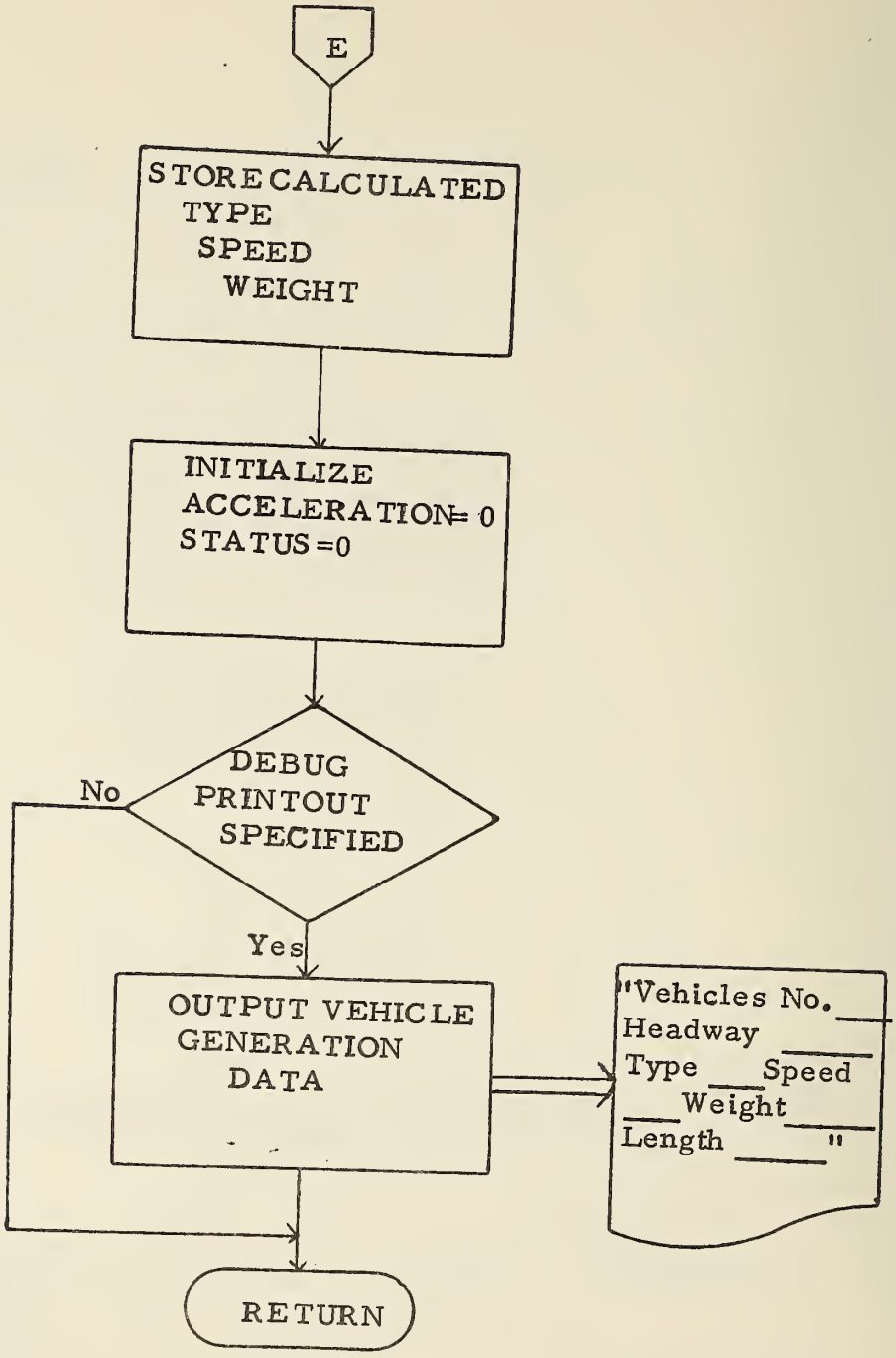


Figure 15. GEN Program Flow Chart (Continued)

```
001              SUBROUTINE GEN                                                  00
          C
          C       GENERATES VEHICLES. CALLS RENUM, HDWAY, TYPE, SPEED, WEIGHT    00
          C      WRITES GENERATED CHARACTERISTICS IF NGEN=1                       00
          C
          C    VEHICLE DATA
002              COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
          C
          C    BRIDGE, ROAD AND TIME DATA
003              COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                 1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                 2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                 3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                 4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
          C
          C    STATISTICAL DATA
004              COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
          C
          C    VEHICLE GENERATION DISTRIBUTION DATA
005              COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
          C
006               INTEGER DISTTY
007               LOGICAL TRUCK, PLATON, DBUG
008               DIMENSION JFWD(200),JBAK(200),JNDX(200)
009               EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
                 1            (JNDX(1),INDX(201))
          C
          C      TOTAL VEHICLES
          C
010               IF (JOKE-1) 900,5,6                                            00
          C
          C      FORWARD
          C
011              5 IF(IFWD(IBAK(1)).EQ.-1)  RETURN
012                GO TO 10
          C
          C REVERSE
          C
013              6 IF(IFWD(JBAK(2)).EQ.-1)  RETURN
014             10 CONTINUE
015                ITV=ITV+1                                                    00
          C
          C  CALCULATE TYPE
          C
016                RNUM=RANF (0.0)                                              01
          C
          C    MODIFY RNUM TO SELECT TRUCK TYPE
017                IF (PLATON(JOKE))  RNUM = RNUM*(1.0-AFR(1,JOKE))+AFR(1,JOKE)
018                DO  30  M=1,MD
```

```
019                ITY = M
020                IF (RNUM.LE.AFR(M,JOKE)) GO TO 40
021           30   CONTINUE                                                  01
022              40 CONTINUE                                                 01
023                IF (PLATON(JOKE))  GO TO 49
024                IF (ITY.EQ.1)  GO TO 49
025                PLATON(JOKE) = .TRUE.
               C
               C   FIND NO TRUCKS IN PLATOON
               C
026                RNUM = RANF(0.0)
027                DO 45 I=1,10
028                IF (RNUM.LE.DPLTON(I,JOKE)) GO TO 46
029                RNUM = RNUM - DPLTON(I,JOKE)
030             45 CONTINUE
031             46 IPLTON(JOKE) = I
               C
               C  DISTRIBUTE GENERATED PLATOONS
               C
032                IGPLTN(I,JOKE) = IGPLTN(I,JOKE) + 1
               C
               C  CALCULATE HEADWAY
               C
033             49 CONTINUE
034                RNUM=RANF (0.0)
035                M=RNUM/DELHD(JOKE)+2                                      00
036                IF(M.GT.LHD(JOKE))M=LHD(JOKE)
037             50 HDINT=HDTAB(M,JOKE)-((HDTAB(M,JOKE)-HDTAB(M-1,JOKE))*
                  1     ((M-1)*DELHD(JOKE)-RNUM))/DELHD(JOKE)
               C
038                IF (.NOT.PLATON(JOKE))  GO TO 81
039                IPLTON(JOKE) = IPLTON(JOKE) - 1
040                IF (IPLTON(JOKE).LE.0)  PLATON(JOKE) = .FALSE.
041             81 CONTINUE
               C
042                IF (.NOT.PLATON(JOKE)) GO TO 51
043                IF (HDINT.GT.2.0)  HDINT = 2.0
044             51 CONTINUE
               C
               C  CALCULATE SPEED
               C
045                RNUM=RANF (0.0)
046                M=RNUM/DELSD(ITY)+2                                       01
047                IF(M.GT.LSP(ITY))M=LSP(ITY)
048                CALCSD=SDTAB(M,ITY) -((SDTAB(M,ITY)-SDTAB(M-1,ITY))*
                  1    ((M-1)*DELSD(ITY)-RNUM))/DELSD(ITY)
049                IF (CALCSD.LT.SPDMIN)  CALCSD = SPDMIN
               C
               C  CALCULATE WEIGHT
               C
050                IF (ITY.NE.1) GO TO 70
051                WEIT=WTAB(1,1)                                            01
052                GO TO 80                                                  01
053             70 RNUM=RANF(0.0)
                                                                             01
```

```
054               M=RNUM/DELWT(ITY)+2
055               IF(M.GT.LWT(ITY)) M=LWT(ITY)
056               WEIT=WTAB(M,ITY)-((WTAB(M,ITY)-WTAB(M-1,ITY))*((M-1)*DELWT(ITY)
                 1   -RNUM))/ DELWT(ITY)
                C
057              80 CONTINUE
                C
058               IF(JOKE.EQ.2)GO TO 20                                          00
059               IBAK(1)=IFWD(IBAK(1))
060               IPV=INDX(IBAK(1))
061               POS(IPV) = -HDFV*CALCSD
062               LANE (IPV)=1                                                   00
063               HDFV = HDFV + HDINT
064               GO TO 21
                C
065              20 JBAK(1)=JFWD(JBAK(1))
066                 IPV=JNDX(JBAK(1))
067                 POS(IPV) = RDEND + HDRV * CALCSD
068                 CALCSD = -CALCSD
069                 LANE(IPV) = -NL
070                 HDRV = HDRV + HDINT
                C
071              21 CONTINUE
072                 ITYPE(IPV)=ITY                                              00
073                 SPD(IPV,1) = CALCSD                                         00
074                 SPD(IPV,2)=CALCSD                                           00
075                 ACC(IPV) = 0.0
076                 WGT(IPV)=WEIT                                               00
077                 KSTAT(IPV) = 0
                C
078             130 IF(.NOT.DBUG) GO TO 140
079                 WRITE (6,100)IPV,HDINT,ITY,CALCSD, WEIT   ,VEHLEN(ITY)
080             100 FORMAT(1H ' VEHICLES NO.',I6,2X,' HEADWAY',F10.3,' TYPE',   00
                   1 I6,' SPEED',F10.3,' WEIGHT',F10.3,' LENGTH',F10.3)         00
081             140 RETURN
                C
082             900 WRITE (6,910) JOKE                                          00
083             910 FORMAT (1H ,'-----ERROR IN NUMBERING',I6)                   00
084                 CALL EXIT                                                   00
085                 END                                                         00
```

GRAPH

The GRAPH subroutine is called UPDATE only if the debug printout is specified. This routine prints out the roadway and vehicle positions up to 6000 ft. If the roadway is less than or equal to 3000 ft, only 3000 ft of roadway is printed. Forward vehicles are shown positionally on the roadway by table index number. Reverse vehicles are shown positionally on the roadway by table index number -150. Reverse vehicle table indices start at 200, and are therefore shown as starting at 50.

No subroutines are called by GRAPH.

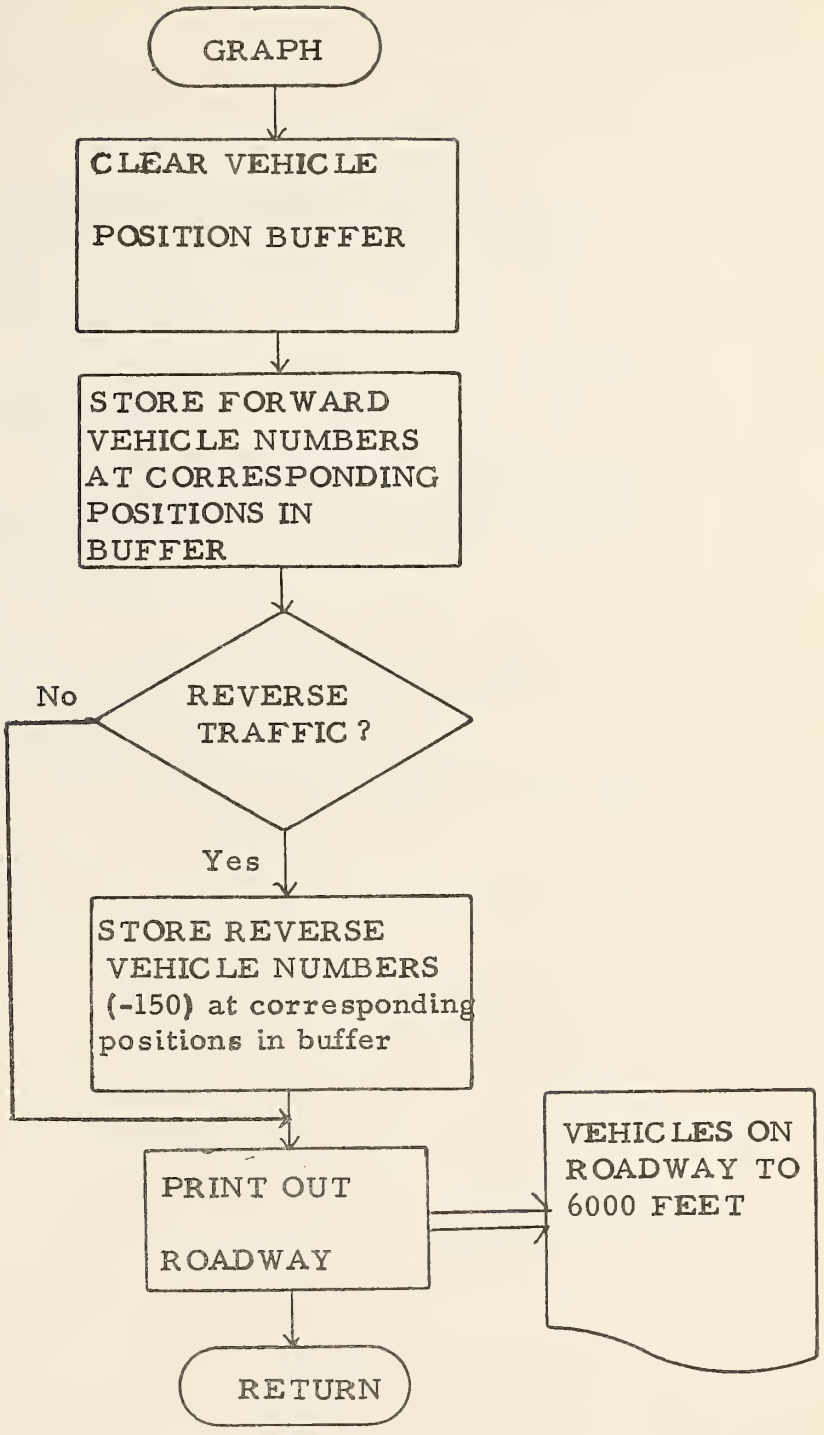


Figure 16. GRAPH Program Flow Chart

```
001             SUBROUTINE GRAPH
                C
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                 1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                 2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                 3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                 4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                 1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                 3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
                C   BRIDGE LOADING DATA
006               COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                 1     XPOS(50), DXPOS(50), ACCLR(50),
                 2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
                C
007               DIMENSION JFWD(200),JBAK(200),JNDX(200)
008               EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
                 1           (JNDX(1),INDX(201))
009               DIMENSION              IVEH(120,2)
                C
010               DO 5 J=1,2
011               DO 5 I=1,120
012             5 IVEH(I,J) = 0
                C
013               IFD = IFWD(1)
014            10 I = INDX(IFD)
015               II = POS(I) * .02
016               J = LANE(I)
                C
017               IF (II.EQ.0)  II = 1
018               IF (II.GT.120)  GO TO 20
                C
019               IVEH(II,J) = I
020            20 IF(IFD.EQ.IBAK(1))  GO TO 30
021               IFD = IFWD(IFD)
022               GO TO 10
                C
023            30 CONTINUE
024               IF (ND.EQ.1)  GO TO 50
025               IFD = JFWD(1)
026            35 I = JNDX(IFD)
```

```
027               II = POS(I) * .02
028               J = -LANE(I)
          C
029               IF (II.EQ.0)  II = 1
030               IF (II.GT.120)  GO TO 40
          C
031               IVEH(II,J) = I - 150
032            40 IF(IFD.EQ.JBAK(1))  GO TO 50
033               IFD = JFWD(IFD)
034               GO TO 35
035            50 CONTINUE
036               WRITE (6,3100)  TOTIM
037               WRITE (6,3120)
038               WRITE (6,3110) (IVEH(I,2),I=1,60)
039               WRITE (6,3121)
040               WRITE (6,3110) (IVEH(I,1),I=1,60)
041               WRITE (6,3120)
          C
042               IF (RDEND.LE.3000.0)  GO TO 110
043               WRITE (6,3101)
044               WRITE (6,3120)
045               WRITE (6,3110) (IVEH(I,2),I=61,120)
046               WRITE (6,3121)
047               WRITE (6,3110) (IVEH(I,1),I=61,120)
048               WRITE (6,3120)
049           110 CONTINUE
050          3100 FORMAT('OSIMULATION TIME =',F10.2,//' ROADWAY 0 TO 3000 FT')
051          3101 FORMAT ('OROADWAY 3000 TO 6000 FT')
052          3110 FORMAT(5X, 60I2)
053          3120 FORMAT(5X,'-----------------------------------------------',
                        1           '-----------------------------------------------',
                        1           '--------------------')
054          3121 FORMAT(5X,'- - - - - - - - - - - - - - - - - - - - '
                        1          ,'- - - - - - - - - - - - - - - - - - - - '
                        1          ,'- - - - - - - - - - - - - - - - - - - - '  )
          C
055               RETURN
056               END
```

INDATA

The INDATA subroutine reads in the following simulation data:

NAMELIST "DATA" input

NTH
TIMLIM
DELTIM
MD
NL
ND
NRAND
IOUT
BRLEN
BRPOS
NZ
SPDLIM
TRKLIM
EXSPD
SPDMIN
ACCEL
SDFAC
SAFDIS
LT
TALINC
DBUG

Tabular data

- Restricted Zones V, W, X, Y, Z
- Vehicle data NOAX, POWER, AXPOS, AXWT
- Acceleration Coefficients FT

All elements (either input or default) are printed out after input. The data SPDLIM, TRKLIM, EXSPD, and SPDMIN are converted from mi/hr to ft/sec before they are printed out.

No subroutines are called by INDATA.

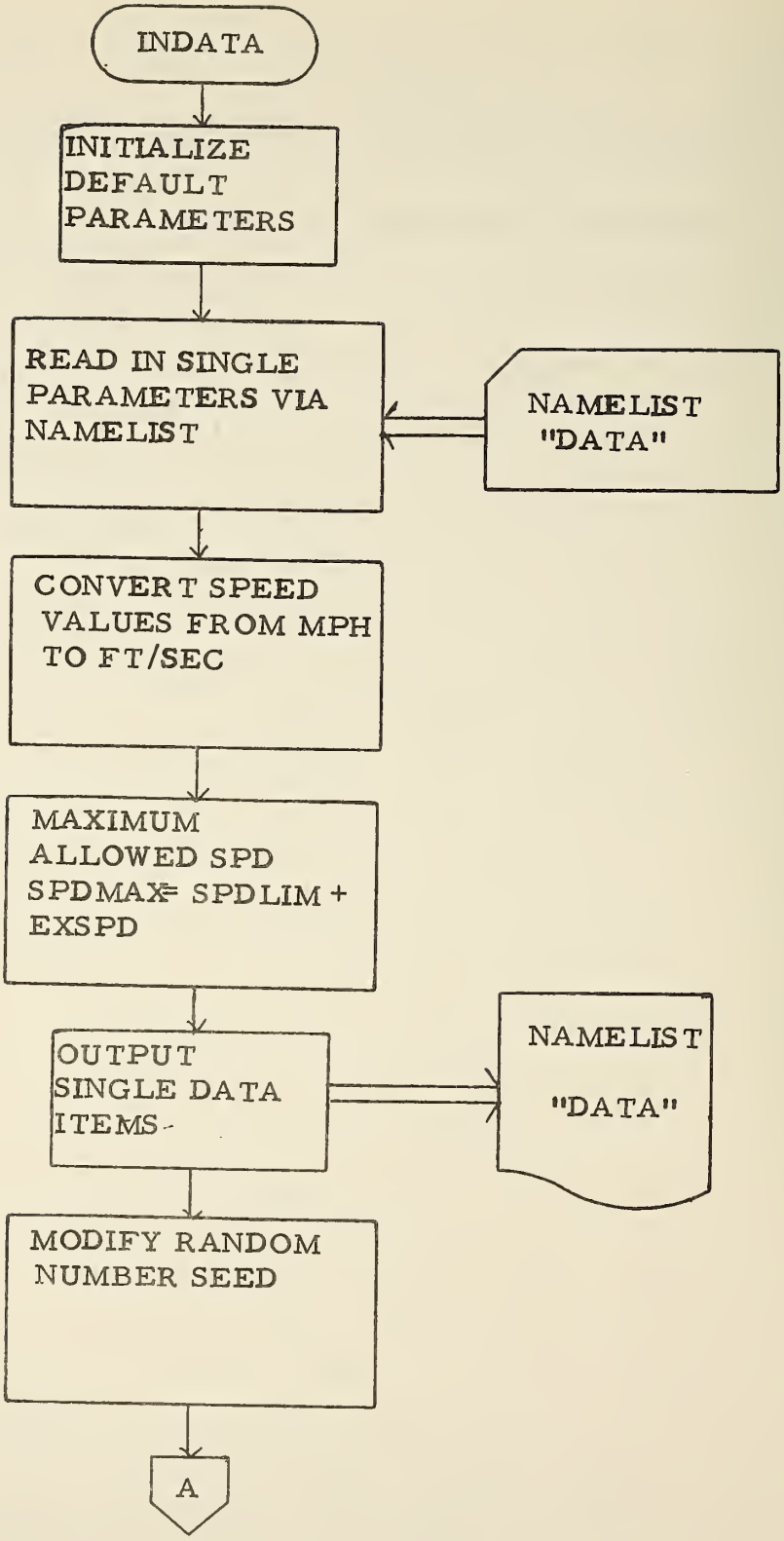


Figure 17. INDATA Program Flow Chart

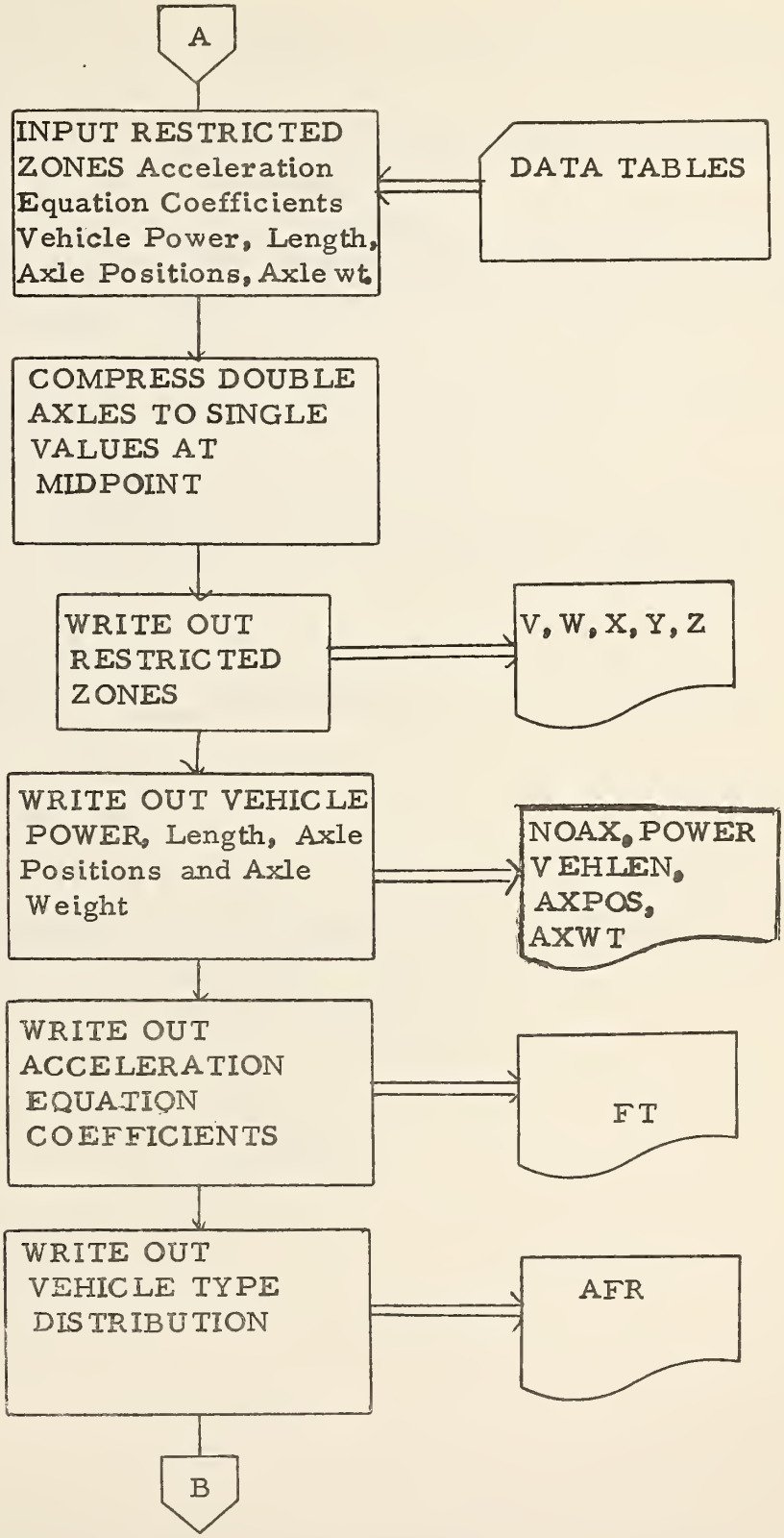


Figure 17. INDATA Program Flow Chart (Continued

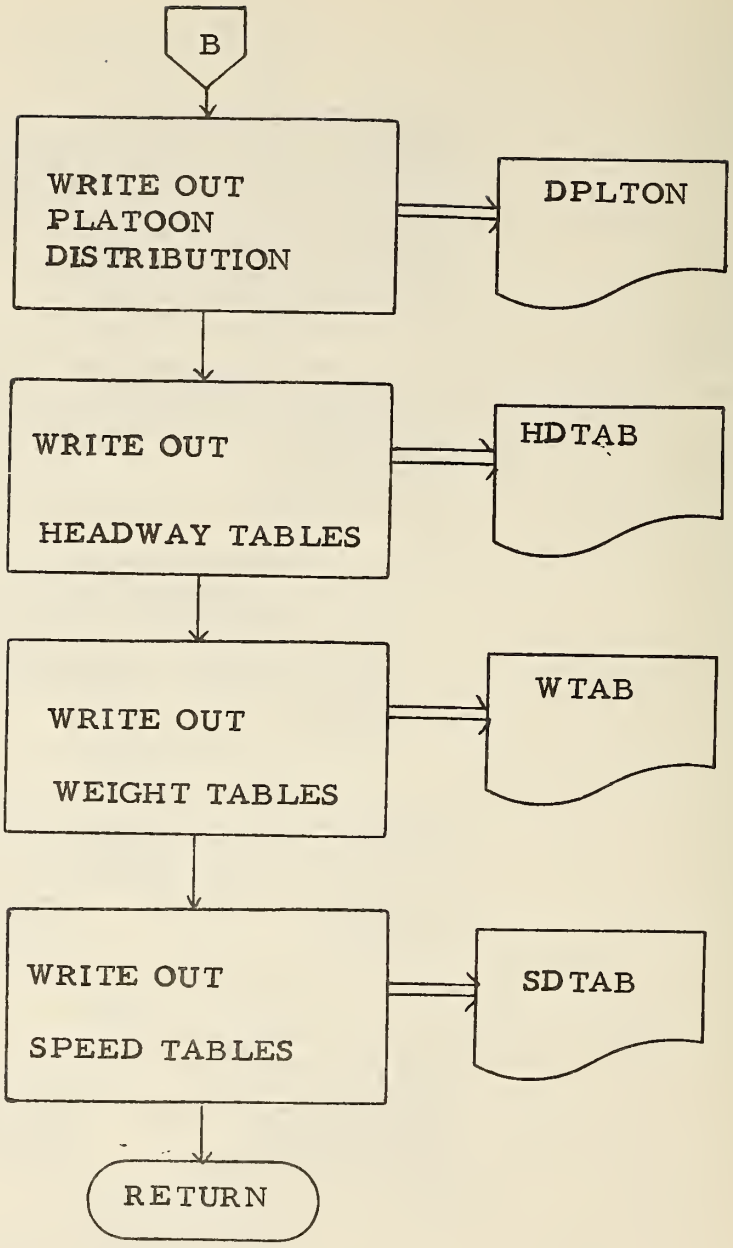


Figure 17. INDATA Program Flow Chart (Continued)

```
001               SUBROUTINE INDATA                                              00
                  C
                  C     INITIALIZES OR READS IN CONSTANT PARAMETERS FOR A SIMULATION      00
                  C
                  C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                  C
                  C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                 1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                 2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                 3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                 4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                  C
                  C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                 1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                  C
                  C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                 3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                  C
                  C   BRIDGE LOADING DATA
006               COMMON /BLK2/ SUMHR, DELTIM, IEVENT, LNUM(50), WEIT(50),
                 1     XPOS(50), DXPOS(50), ACCLR(50),
                 2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
                  C
007               COMMON /RANDOM/ IX,IY,YFL
                  C
008               LOGICAL DBUG, FIRST
                  C
009               DATA NAM1,NAM2,NAM3,NAM4     /4HEND ,4HZONE,4HCOEF,4HVEHI/
                  C
010               NAMELIST/DATA/ NTH, TIMLIM, DELTIM, MD,NL,ND, NRAND, IOUT, BRLEN,
                 1     BRPOS, NZ, SPDLIM, TRKLIM, EXSPD, SPDMIN, ACCEL, SDFAC,
                 2     SAFDIS, LT, TALINC, DBUG
                  C
                  C       THIS SECTION ESTABLISHES DEFAULT VALUES
011               NTH=1
012               MD=11
013               LT=12
014               NL=2
015               ND=1
016               NZ=0
017               TIMLIM=1.0
018               DELTIM=1.0
019               BRPOS=1100.
020               TALINC=8000.
021               ACCEL=15.0
022               SPDMIN=40.0
023               SPDLIM= 65.0
```

```
024               EXSPD=15.0
025               TRKLIM= 55.0
026               IOUT = 2
027               BRLEN=100.
028               NRAND = 0
029               DBUG = .FALSE.
030               DO 11 I=1,5
031               V(I)=0.0
032               W(I)=0.0
033               X(I)=0.0
034               Y(I)=0.0
035               Z(I)=0.0
036            11 CONTINUE
                C
037             1000 FORMAT (A4,6X,I2)
038             1001 FORMAT (48H JOB ABORTED-CARDS IMPROPERLY SEQUENCED COL 1-4 ,A4,
                    112H  COL 10-11 ,A2)
039             1002 FORMAT (4(F7.1),F5.2)
040             1003 FORMAT (4(F10.5))
041             1004 FORMAT (A4,20(2X,I1))
042             1005 FORMAT (I1,F5.0,F4.1,5(F4.1,F3.3))
043             2000 FORMAT (1H0)
044             2001 FORMAT (1H1)
045             2002 FORMAT(58X,10HZONE  DATA)
046             2003 FORMAT(11X,21HBEGIN FORWARD UPGRADE,3X,19HEND FORWARD UPGRADE,3X,
                    121HBEGIN REVERSE UPGRADE,3X,19HEND REVERSE UPGRADE,3X,16HPERCENT O
                    2F GRADE)
047             2004 FORMAT(17X,F7.1,17X,F7.1,16X,F7.1,15X,F7.1,15X,F5.2)
048             2005 FORMAT(56X,13HVEHICLE  DATA)
049             2006 FORMAT(6X,12HVEHICLE TYPE,13X,20(I2,3X))
050             2007 FORMAT(6X,15HNUMBER OF AXLES,11X,20(I1,4X))
051             2008 FORMAT(6X,13HVEHICLE POWER,10X,20(F4.0,1X))
052             2009 FORMAT(6X,14HVEHICLE LENGTH,9X,20(F4.1,1X))
053             2010 FORMAT(6X,19HFIRST AXLE POSITION,4X,20(F4.1,1X))
054             2011 FORMAT(6X,22HPERCENT WEIGHT ON AXLE,2X,20(F3.2,2X))
055             2012 FORMAT(6X,20HSECOND AXLE POSITION,3X,20(F4.1,1X))
056             2013 FORMAT(6X,19HTHIRD AXLE POSITION,4X,20(F4.1,1X))
057             2014 FORMAT(6X,20HFOURTH AXLE POSITION,3X,20(F4.1,1X))
058             2015 FORMAT(6X,19HFIFTH AXLE POSITION,4X,20(F4.1,1X))
059             2016 FORMAT(51X,30HCOEFFICIENTS  OF  ACCELERATION)
060             2017 FORMAT(21X,17HWEIGHT/HORSEPOWER,8X,47HC(0)   +   C(1)V   +   C(2)V*
                    1*2 + C(3)TAN(THETA))
061             2018 FORMAT(26X,7H  0-50 ,10X,4(F10.5,2X))
062             2019 FORMAT(26X,7H 50-100,10X,4(F10.5,2X))
063             2020 FORMAT(26X,7H100-200,10X,4(F10.5,2X))
064             2021 FORMAT(26X,7H200-300,10X,4(F10.5,2X))
065             2022 FORMAT(26X,7H300-400,10X,4(F10.5,2X))
066             2023 FORMAT(25X,8HOVER 400,10X,4(F10.5,2X))
067                  READ (5,DATA)
068                  IF (LT.GT.50) LT = 50
069                  SPDMIN=SPDMIN * 1.46667
070                  SPDLIM=SPDLIM * 1.46667
071                  EXSPD=EXSPD * 1.46667
072                  TRKLIM=TRKLIM * 1.46667
```

```
073               SPDMAX = SPDLIM + EXSPD
                  C
                  C   MODIFY RANDOM NUMBER SEED
                  C
074               IX = IX + NRAND * IX
                  C
075             100 READ (5,1000) NAM0,INT1
076                 IF (NAM0.EQ.NAM1) GO TO 500
077                 IF (NAM0.EQ.NAM2) GO TO 200
078                 IF (NAM0.EQ.NAM3) GO TO 300
079                 IF (NAM0.EQ.NAM4) GO TO 400
080             150 WRITE (6,1001) NAM0,INT1
081                 GO TO 600
082             200 NZ=INT1
083                 DO 210 I=1,INT1
084                 READ (5,1002) V(I),W(I),X(I),Y(I),Z(I)
085             210 CONTINUE
086                 GO TO 100
087             300 DO 310 I=1,6
088                 READ (5,1003) (FT(J,I),J=1,4)
089             310 CONTINUE
090                 GO TO 100
                  C
091             400 MD=INT1
092                 DO 420 I=1,INT1
093                 READ (5,1005) NOAX(I),
                   1              POWER(I),VEHLEN(I),(AXPOS(J,I),AXWT(J,I),J=1,5)
094             420 CONTINUE
095                 J=INT1+1
096                 DO 440 I=J,20
097                 NOAX(I)=0
098                 POWER(I)=0.0
099                 VEHLEN(I)=0.0
100                 DO 430 K=1,5
101                 AXPOS(K,I)=0.0
102                 AXWT(K,I)=0.0
103             430 CONTINUE
104             440 CONTINUE
105                 DO 460 I=1,INT1
106                 DO 450 K=1,5
107                 IF (AXPOS(K,I).EQ.0.0) GO TO 450
108                 J=K+1
109                 IF (J.GT.5) GO TO 450
110                 IF ((AXPOS(J,I)-AXPOS(K,I)).GT.(5.0)) GO TO 450
111                 IF ((AXPOS(J,I)-AXPOS(K,I)).LT.(0.0)) GO TO 450
112                 AXPOS(K,I)=(AXPOS(K,I)+AXPOS(J,I))/2.0
113                 AXWT(K,I)=(AXWT(K,I)+AXWT(J,I))
114                 NOAX(I)=NOAX(I)-1
115                 DO 446 L=J,4
116                 M=L+1
117                 AXPOS(L,I)=AXPOS(M,I)
118                 AXWT(L,I)=AXWT(M,I)
119             446 CONTINUE
120                 AXPOS(5,I)=0.0
```

```
121             AXWT(5,I)=0.0
122           450 CONTINUE
123           460 CONTINUE
124               GO TO 100
          C
125           500 CONTINUE
126               DO 470 J=1,2
127               DO 470 I=1,20
128           470 AFR(I,J) = AFS(I,J)
          C
129               WRITE(6,DATA)
130               WRITE(6,2001)
131               WRITE(6,2000)
132               WRITE(6,2000)
133               WRITE(6,2002)
134               WRITE(6,2000)
135               WRITE(6,2003)
136               DO 510 I=1,5
137               WRITE(6,2004) V(I),W(I),X(I),Y(I),Z(I)
138           510 CONTINUE
139               WRITE(6,2000)
140               WRITE(6,2000)
141               WRITE(6,2000)
142               WRITE(6,2005)
143               WRITE(6,2000)
144               WRITE(6,2006) (I,I=1,20)
145               WRITE(6,2000)
146               WRITE(6,2007) (NOAX(I),I=1,20)
147               WRITE(6,2008) (POWER(I),I=1,20)
148               WRITE(6,2009) (VEHLEN(I),I=1,20)
149               WRITE(6,2000)
150               WRITE(6,2010) (AXPOS(1,I),I=1,20)
151               WRITE(6,2011) (AXWT(1,I),I=1,20)
152               WRITE(6,2000)
153               WRITE(6,2012) (AXPOS(2,I),I=1,20)
154               WRITE(6,2011) (AXWT(2,I),I=1,20)
155               WRITE(6,2000)
156               WRITE(6,2013) (AXPOS(3,I),I=1,20)
157               WRITE(6,2011) (AXWT(3,I),I=1,20)
158               WRITE(6,2000)
159               WRITE(6,2014) (AXPOS(4,I),I=1,20)
160               WRITE(6,2011) (AXWT(4,I),I=1,20)
161               WRITE(6,2000)
162               WRITE(6,2015) (AXPOS(5,I),I=1,20)
163               WRITE(6,2011) (AXWT(5,I),I=1,20)
164               WRITE(6,2001)
165               WRITE(6,2000)
166               WRITE(6,2000)
167               WRITE(6,2000)
168               WRITE(6,2016)
169               WRITE(6,2000)
170               WRITE(6,2017)
171               WRITE(6,2018) (FT(I,1),I=1,4)
172               WRITE(6,2019) (FT(I,2),I=1,4)
```

```
      WRITE(6,2020) (FT(I,3),I=1,4)
      WRITE(6,2021) (FT(I,4),I=1,4)
      WRITE(6,2022) (FT(I,5),I=1,4)
      WRITE(6,2023) (FT(I,6),I=1,4)
  700 WRITE(6,3001)
      WRITE(6,3000)
      WRITE(6,3002)
      WRITE(6,3000)
      WRITE(6,3003) (I,I=1,20)
      WRITE(6,3004)
      DO 710 J=1,2
      WRITE(6,3005) J,(AFR(I,J),I=1,20)
  710 CONTINUE
      WRITE(6,3000)
      WRITE(6,3000)
      WRITE(6,3000)
      WRITE(6,3006)
      WRITE(6,3000)
      WRITE(6,3070) (I,I=1,10)
      WRITE(6,3004)
      DO 720 J=1,2
      WRITE(6,3050) J,(DPLTON(I,J),I=1,10)
  720 CONTINUE
      WRITE(6,3000)
      WRITE(6,3000)
      WRITE(6,3000)
      WRITE(6,3008)
      WRITE(6,3009) (I,I=1,20)
      WRITE(6,3004)
      J=1
      WRITE(6,3010) J,(HDTAB(I,1),I=1,20)
      J=2
      WRITE(6,3010) J,(HDTAB(I,2),I=1,20)
      WRITE(6,3000)
      WRITE(6,3009) (I,I=21,40)
      WRITE(6,3004)
      J=1
      WRITE(6,3010) J,(HDTAB(I,1),I=21,40)
      J=2
      WRITE(6,3010) J,(HDTAB(I,2),I=21,40)
      WRITE(6,3001)
      WRITE(6,3011)
      WRITE(6,3030) (I,I=1,12)
      WRITE(6,3013)
      DO 730 J=1,30
      WRITE(6,3414) J,(WTAB(J,I),I=1,12)
  730 CONTINUE
      IF(MD.LT.13)  GO TO 736
      WRITE(6,3001)
      WRITE(6,3011)
      WRITE(6,3030) (I,I=13,20)
      WRITE(6,3013)
      DO 735 J=1,50
      WRITE(6,3414) J,(WTAB(J,I),I=13,20)
```

```
227          735 CONTINUE
228          736 CONTINUE
229              WRITE(6,3000)
230              WRITE(6,3000)
231              WRITE(6,3012)
232              WRITE(6,3003) (I,I=1,20)
233              WRITE(6,3013)
234              DO 740 J=1,20
235              WRITE(6,3014)J,(SDTAB(J,I),I=1,20)
236          740 CONTINUE
237         3000 FORMAT(1H0)
238         3001 FORMAT(1H1)
239         3002 FORMAT(59X,20HTRAFFIC DISTRIBUTION)
240         3003 FORMAT(2X,12HVEHICLE TYPE,3X,I2,19(4X,I2))
241         3030 FORMAT(2X,12HVEHICLE TYPE,3X,I2,11(6X,I2))
242         3004 FORMAT(4X,9HDIRECTION)
243         3005 FORMAT(7X,I2,5X,F5.3,19(1X,F5.3))
244         3050 FORMAT(7X,I2,5X,F5.3, 9(2X,F6.3))
245         3006 FORMAT(55X,26HTRUCK PLATOON DISTRIBUTION)
246         3007 FORMAT(1X,16HNUMBER OF TRUCKS,I2,19(4X,I2))
247         3070 FORMAT(1X,16HNUMBER OF TRUCKS,I2, 9(6X,I2))
248         3008 FORMAT(61X,15HHEADWAY  TABLES)
249         3009 FORMAT(2X,12HVALUE NUMBER,3X,I2,19(4X,I2))
250         3010 FORMAT(7X,I2,5X,F5.2,19(1X,F5.2))
251         3011 FORMAT(59X,14HWEIGHT  TABLES)
252         3012 FORMAT(59X,14HSPEED   TABLES)
253         3013 FORMAT(6X,5HVALUE)
254         3014 FORMAT(7X,I2,5X,F5.0,19(1X,F5.0))
255         3414 FORMAT(7X,I2,3X,F7.0,11(1X,F7.0))
256          600 RETURN
257              END
```

INIT

The INIT subroutine initializes buffer allocation tables and the following simulation parameters:

Simulation time
Total vehicles
Forward headway
Reverse headway
Platoon pointers and distribution
No trucks this event
Event number
Load distribution - sampled
Type distribution - sampled

No subroutines are called by INIT.

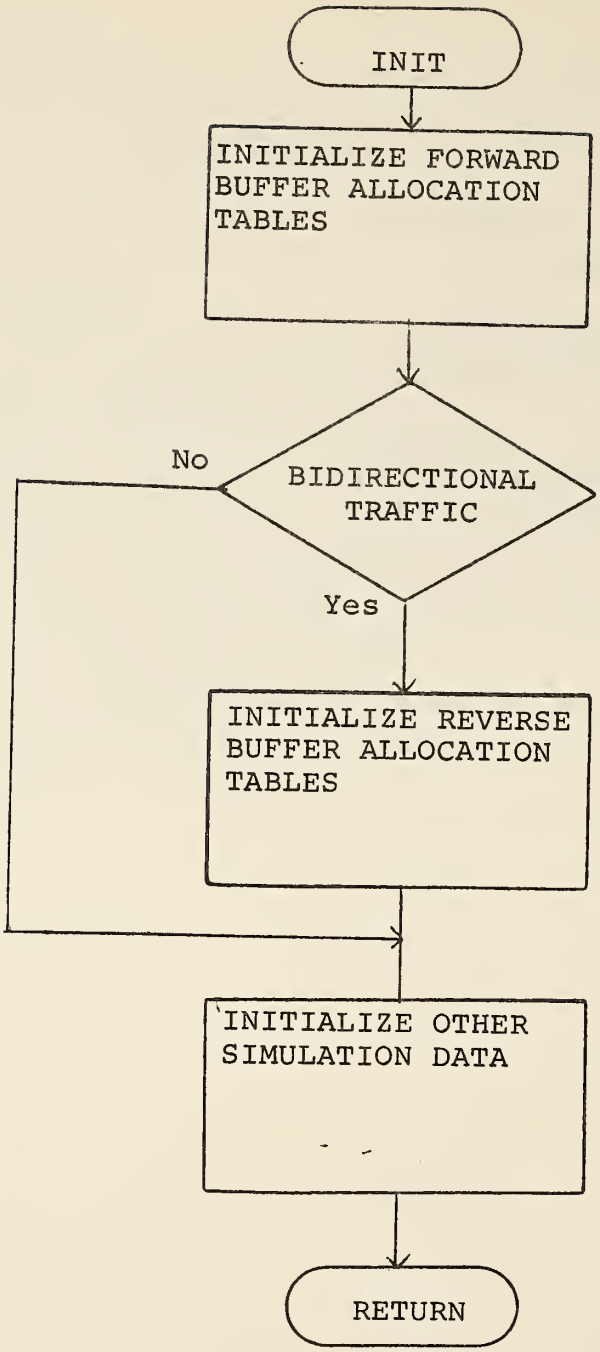


Figure 18. INIT Program Flow Chart

```
001               SUBROUTINE INIT
                  C
                  C  THIS ROUTINE INITIALIZES VEHICLE DATA BUFFER ALLOCATION AND FORWARD /
                  C           BACKWARD LINKS
                  C
                  C   VEHICLE DATA
002                 COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                   1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                  C
                  C   BRIDGE, ROAD AND TIME DATA
003                 COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                   1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                   2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                   3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                   4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                  C
                  C   STATISTICAL DATA
004                 COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                   1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                  C
                  C   VEHICLE GENERATION DISTRIBUTION DATA
005                 COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                   1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                   2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                   3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                  C
006                 COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                   1     XPOS(50), DXPOS(50), ACCLR(50),
                   2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
                  C
007                 COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
                   1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
                   2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
                  C
008                 INTEGER DISTTY, DISTLD
009                 LOGICAL DATABL, FIRST, LAST
010                 LOGICAL PLATON, TRUCK
011                 DIMENSION JFWD(200),JBAK(200),JNDX(200)
012                 EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
                   1            (JNDX(1),INDX(201))
                  C
013                 DO 100 I=1,400
014                 IFWD(I)=I+1
015                 IBAK(I)=I-1
016               100 INDX(I)=I-1
017                 IFWD(400) = -1
018                 IBAK(1)=1
019                 IF(ND.EQ.1) GO TO 300
                  C
020                 IFWD(200) = -1
021                 DO 200 I=1,200
022                 JFWD(I)=I+1
023               200 JBAK(I)=I-1
024                 JFWD(200) = -1
```

```
025                JBAK(1)=1
                C
026              300 CONTINUE
027                  MPLTON = 0
028                  TOTIM=0.0
029                  ITV=0                                                  00
030                  JOK=0                                                  00
031                  HDFV=0.                                                00
032                  HDRV=0.                                                00
033                  PLATON(1) = .FALSE.                                    00
034                  PLATON(2) = .FALSE.
035                  FIRST = .TRUE.
036                  IEVENT = 0
037                  TRUCK = .FALSE.
038                  LAST = .FALSE.
039                  NTRUK = 0
040                  IEVENT = 0
041                  NTRUKX = 0
042                  DO 500 I=1,10
043                  IDPLTN(I)=0
044                  DO 500 J=1,2
045              500 IGPLTN(I,J)=0
046                  DO 600 I=1,50
047              600 DISTLD(I)=0
048                  DO 700 I=1,20
049              700 DISTTY(I)=0
050                  RETURN
051                  END
```

ORDER

This subroutine is called by UPDATE whenever there is bridge loading data to be output. Every call but the first one, the previously ordered data block is written out with the LAST switch set "FALSE" if the current event number is the same as the data block event number, or set "TRUE" if the current event number is different indicating that this is the last data block of this event. The new axle load data is then ordered positionally and stored for output next time the ORDER subroutine is called. If the DBUG switch is set "TRUE" a printout of the axle load data and truck identification data is generated.

No subroutines are called by the ORDER subroutine.

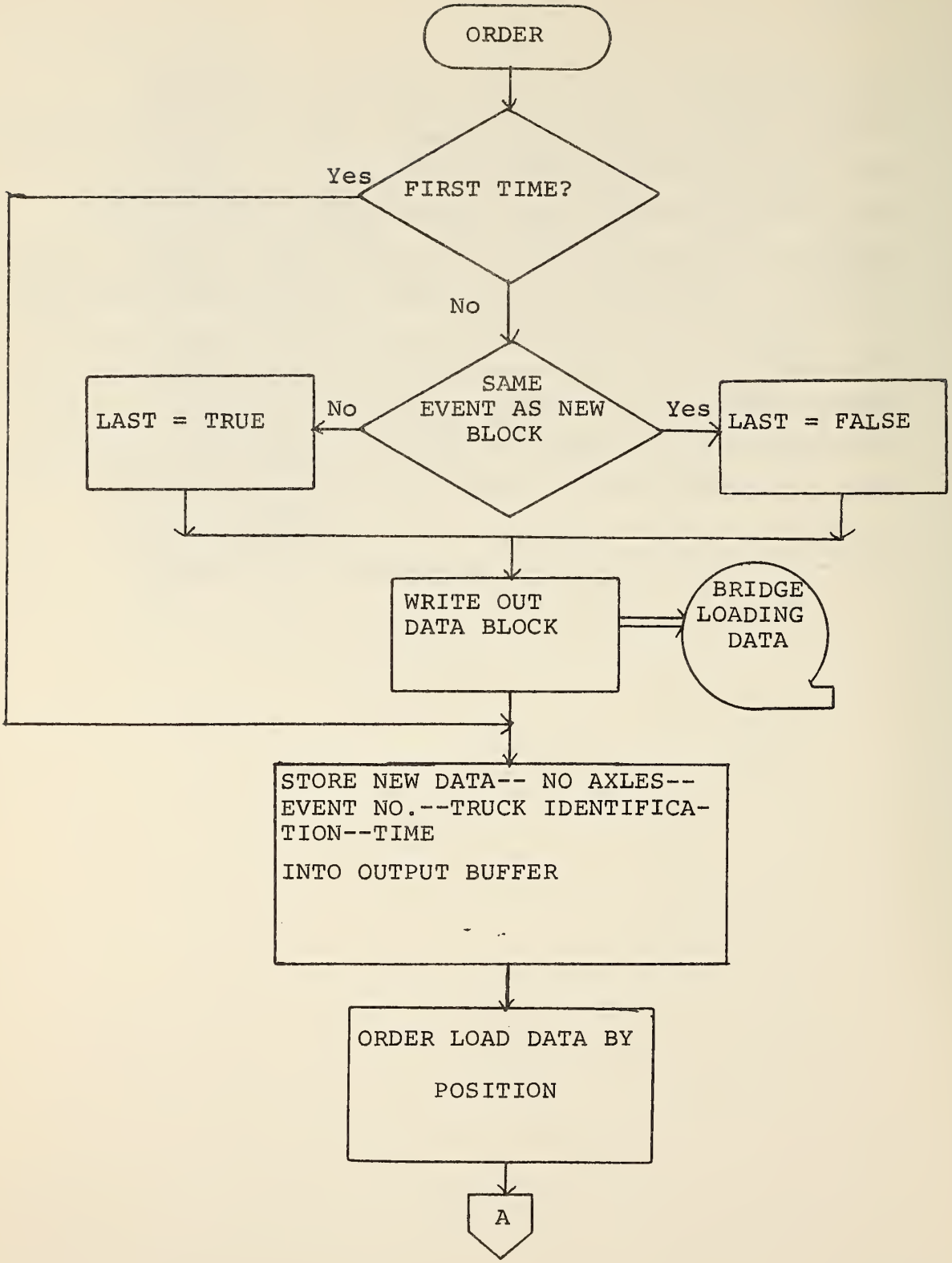


Figure 19. ORDER Program Flow Chart

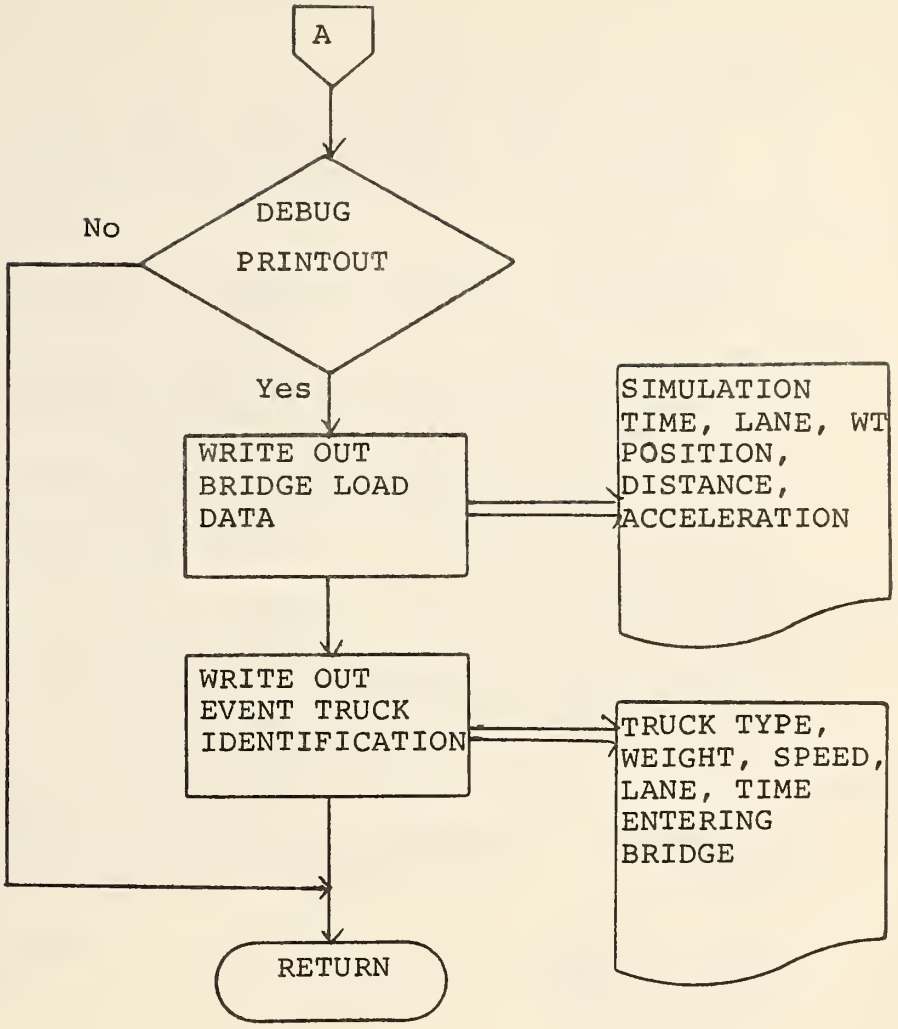


Figure 19. ORDER Program Flow Chart (Continued)

```
001             SUBROUTINE ORDER
                C
002             COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
               1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003             COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004             COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   BRIDGE LOADING DATA
005             COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                1     XPOS(50), DXPOS(50), ACCLR(50),
                2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
006             COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
                1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
                2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
007             DIMENSION LOADBL(376)
008             EQUIVALENCE(LOADBL(1),SUMHRX)
009             LOGICAL DBUG, LAST
010             DATA IFRST/0/
                C
011             IF (IFRST.EQ.0)  GO TO 210
012             LAST = .FALSE.
013             IF (IEVENT.NE.IEVNTX)  LAST = .TRUE.
                C
                C
                C                                                    )4A02( TAMROF
014             WRITE(IOUT)       LOADBL                        LBDAOL  )0021,TUOI(ETIRW
015           210 CONTINUE                                                    00
016             IFRST = 1
                C
017             NOAXLX = NOAXL
018             NEVENT = IEVNTX
019             IEVNTX = IEVENT
020             DTL = DELTIM
021             K = NOAXL
022             IF (NTRUK.EQ.NTRUKX .AND.IEVENT.EQ.NEVENT)  GO TO 6
023             L1 = 1
024             IF (IEVENT.EQ.NEVENT)  L1 = NTRUKX + 1
025             DO 5 I=L1,NTRUK
026             LTYPE(I) = KTYPE(I)
027             WGTX(I) = WEIT(I)
028             SPDX(I) = SPDT(I)
029             LLANE(I) = KLANE(I)
030           5 TIMEX(I) = TIMET(I)
031             NTRUKX = NTRUK
032           6 SUMHRX = SUMHR
```

```
033               KK = K
                  C
                  C   ORDER THE DATA BY POSITION
                  C
034               DO 20 II=1,K
035               J = 1
036               POSOLD = XPOS(1)
                  C
037               DO 10 I = 1,K
038               IF(XPOS(I).GE.POSOLD)  GO TO 10
039               J = I
040               POSOLD = XPOS(I)
041            10 CONTINUE
                  C
042               XPOSX(II) = XPOS(J)
043               LNUMX(II) = LNUM(J)
044               WEITX(II) = WEIT(J)
045               DXPOSX(II) = DXPOS(J)
046               ACCLRX(II) = ACCLR(J)
047               XPOS(J) = 500000.0
                  C
048            20 CONTINUE
                  C
049               IF(.NOT. DBUG) GO TO 148
050               WRITE(6,1145) TOTIM
051               DO 146 I=1,K
052           146 WRITE(6,1146) I,LNUMX(I), WEITX(I),XPOSX(I),DXPOSX(I),ACCLRX(I)
                  C
053           1145 FORMAT('OBRIDGE LOAD TIME =',F12.4,'SEC'//10X,'LANE',4X,'WEIGHT  '
                 1,'POSITION',2X,'DISTANCE',2X,'ACCELERATION  ORDER'/)
054           1146 FORMAT(7X,I2,3X,I2,F10.0,F10.2,F10.2,F14.2)
055               WRITE (6,1050) IEVENT,NTRUK
056           1050 FORMAT ('0EVENT NUMBER',I4,8X,'NO. OF TRUCKS',I4//,
                 1  1X, 'TYPE    WEIGHT   SPEED    LANE    TIME ENTERING BRIDGE')
057               DO 151 I=1,NTRUK
058           151 WRITE (6,1051) KTYPE(I),WGTT(I),SPDT(I),KLANE(I),TIMET(I)
059           1051 FORMAT(1X,I4,F10.0,F8.0,I8,F8.0)
060           148 CONTINUE
                  C
061               RETURN
062               END
```

PASPOS

The PASPOS subroutine is called by UPDATE to determine whether a maneuvering vehicle may pass its forward vehicle.

On two lane rural highways, vehicles are not permitted to pass, if the lead car or oncoming car is passing, or if the vehicle is in a curve. There must not be less than three lengths of the maneuvering vehicle between the lead car and its lead car.

$$x_{Lt} - x_{Ft} \geq 3\ H_M \tag{2}$$

If these conditions are satisfied, then the time required for the maneuvering vehicle to pass is calculated as follows:

$$x'_M = x_{Mo} + v_M\ T_r + 1/2\ a_M T_r^2 \tag{3}$$

This is the extrapolated position of the maneuvering vehicle based on an acceleration using equation 3 and its present speed.

$$x'_L = x_{Lo} + v_L\ T_r \tag{4}$$

The extrapolated position of the lead vehicle does not include an acceleration term. In the time T_r, the maneuvering vehicle must pass the lead vehicle by its own length plus a minimum distance.

$$x'_M = x'_L + H_M + D \tag{5}$$

T_r is obtained by substituting equations (3) and (4) into (5) and solving the quadratic. In the time T_r, v_M must not exceed the maximum speed permitted. If it does, a new value of T_r is determined using:

$$x'_M = x_{Mo} + 1/2\ T_1\ (v_{Mo} + v_{max}) + v_{max} T_2 \qquad (6)$$

where the new $T_r = T_1 + T_2$.

If the required time for passing exceeds 30 seconds, passing is prohibited; the maneuvering vehicle must follow and no further tests are made. If T_r is less than 30 seconds, the extrapolated position of the oncoming vehicle is calculated

$$x'_A = x_{Ao} + v_{Ao}\ T_r \qquad (7)$$

Now the criteria is

$$x'_M \leq x'_A - 2H_M \qquad (8)$$

that is, the extrapolated position of the maneuvering vehicle must lack two vehicle lengths of collision with the oncoming vehicle. If the last test is passed the vehicle is considered in the passing state.

On multilane highways, passing is much simpler. The maneuvering vehicle cannot be in the leftmost lane. If this test is satisfied, the value of the lag and the lead are examined. The following criteria must all be satisfied to permit the vehicle to move left.

$$\text{Lag} \geq 40 \text{ feet}$$

Lead $\geq$ 30 feet

Lag + Lead $\geq$ 150 feet

If it is determined that the vehicle may pass, its status is set to +1 and the lane is changed to the left lane.

The following subroutines are called by PASPOS:

CALACC

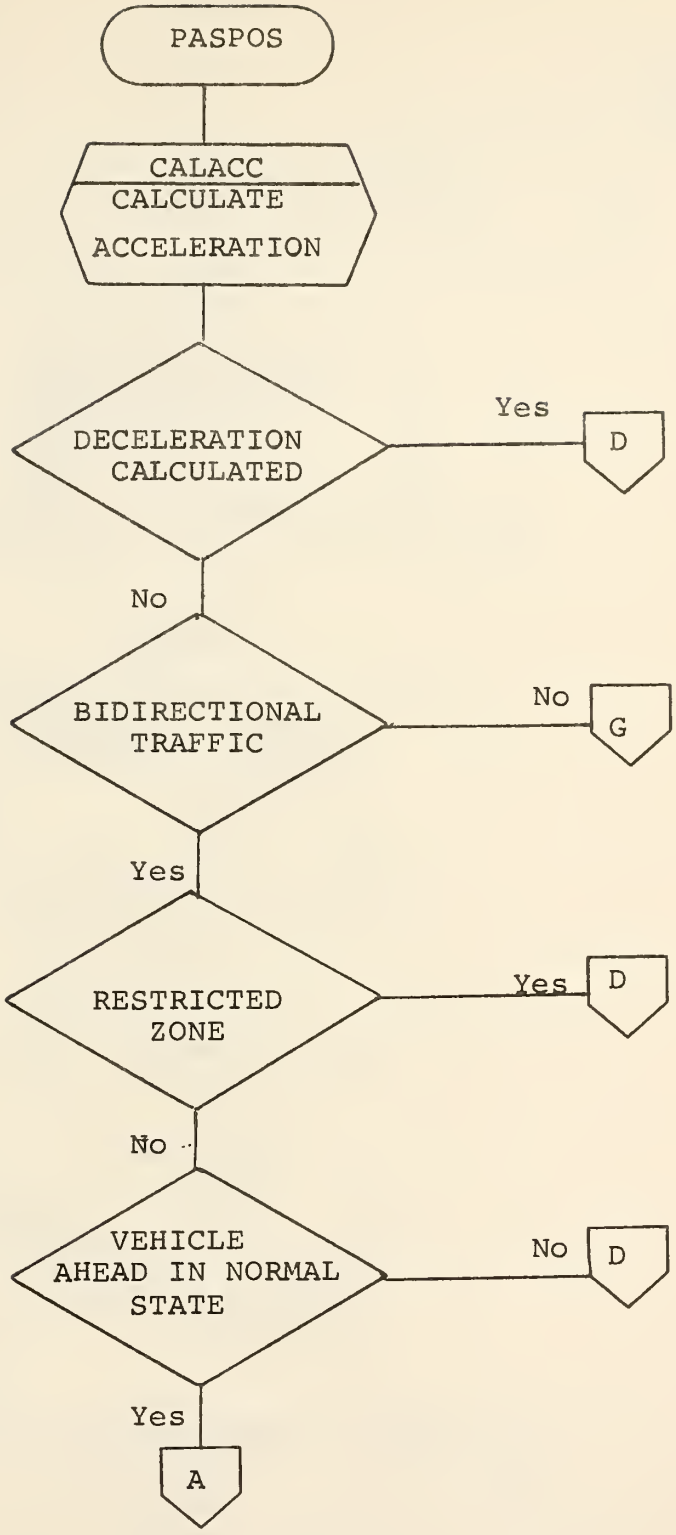


Figure 20. PASPOS Program Flow Chart

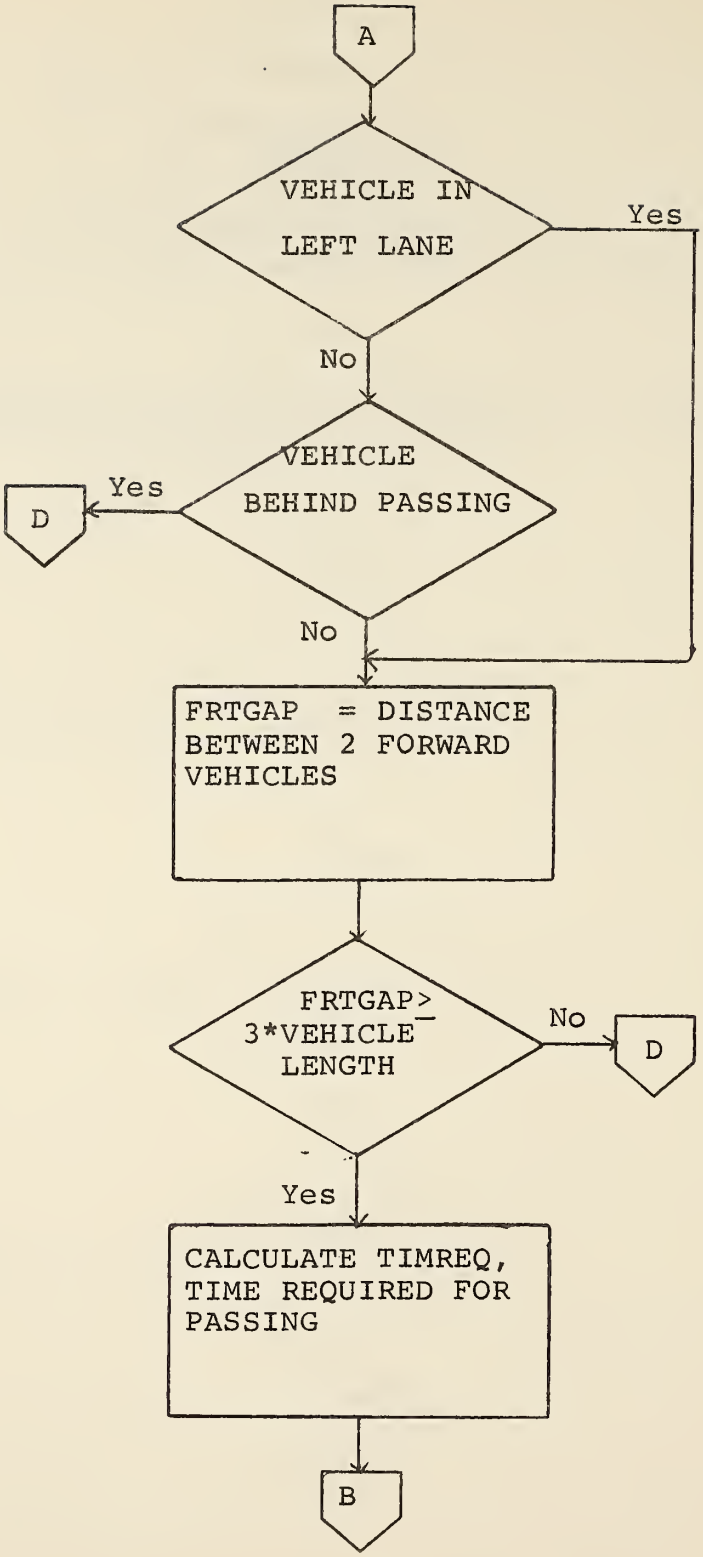


Figure 20. PASPOS Program Flow Chart (Continued)

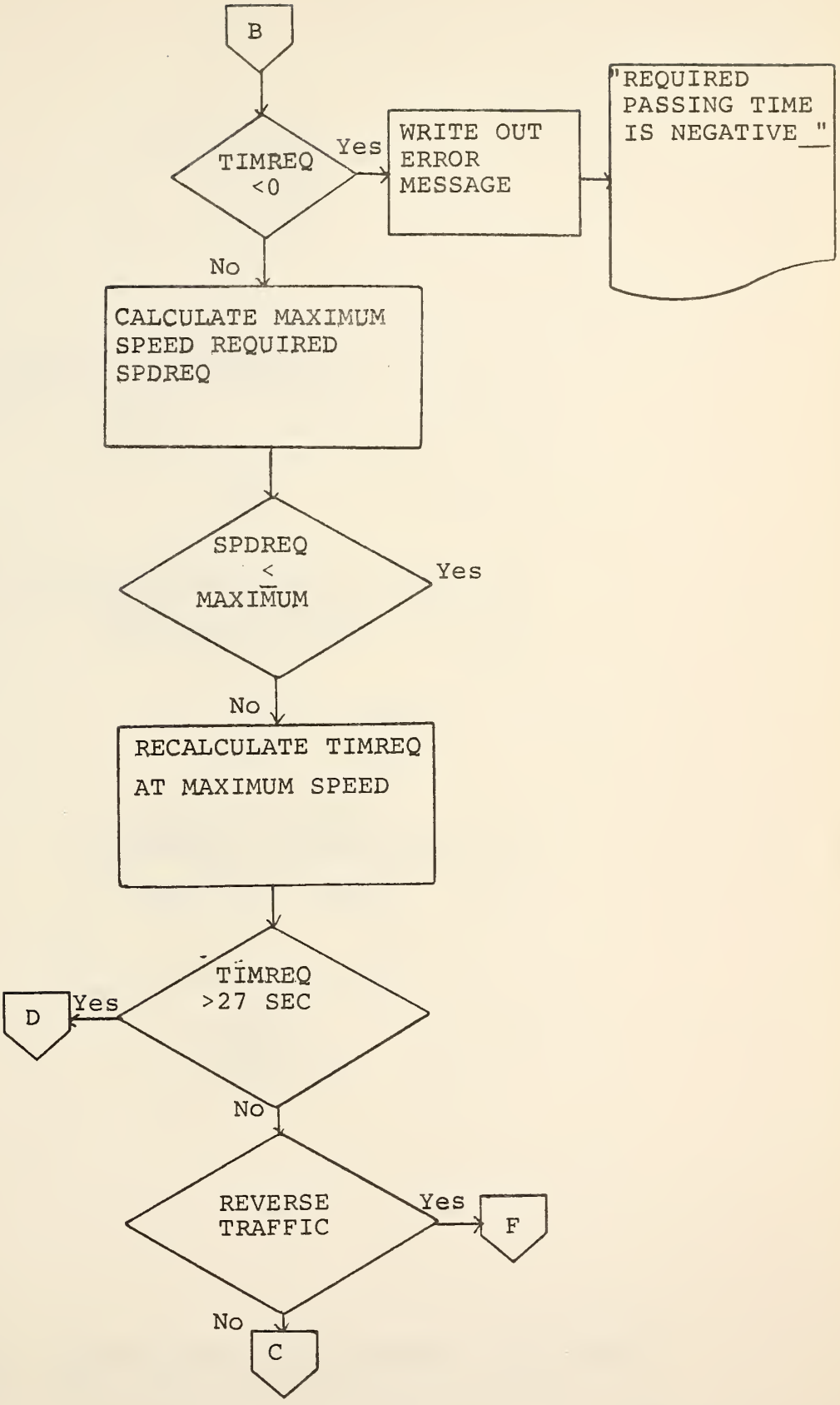


Figure 20. PASPOS Program Flow Chart (Continued)

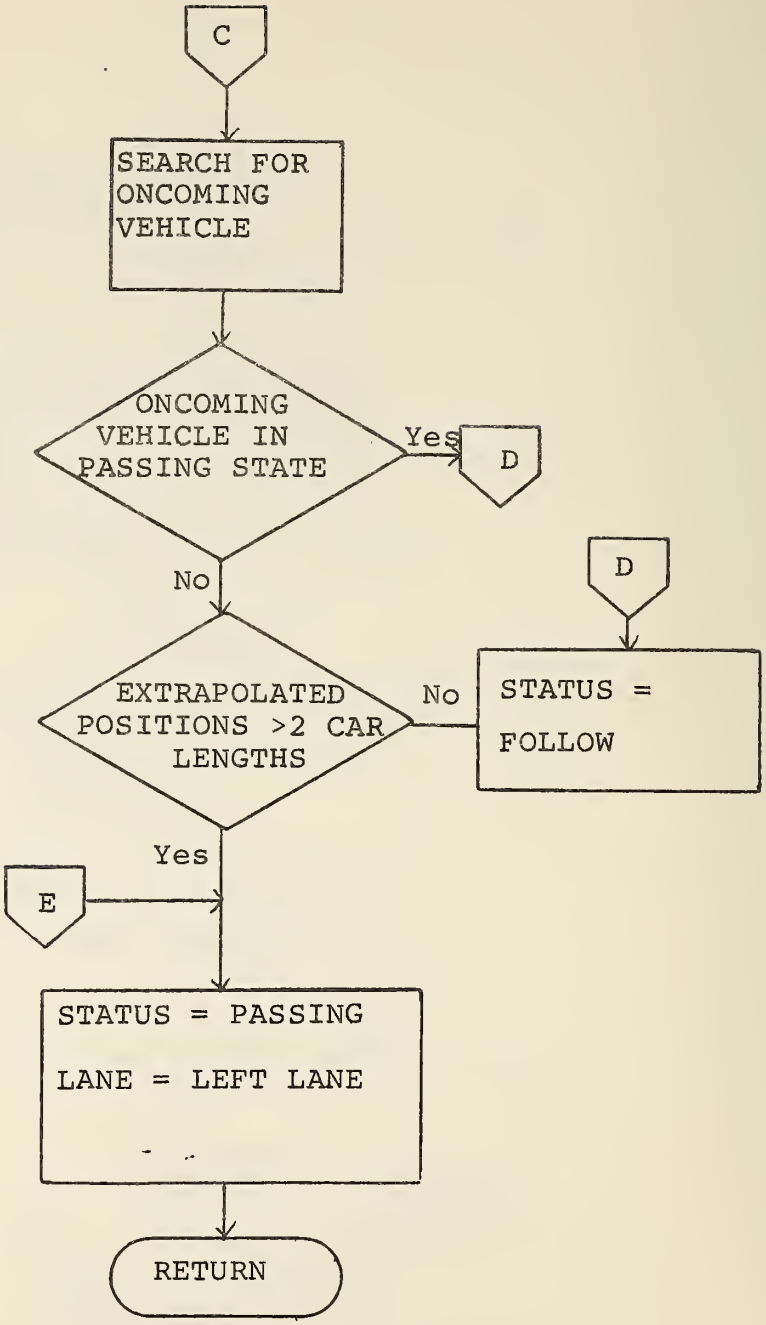


Figure 20. PASPOS Program Flow Chart (Continued)

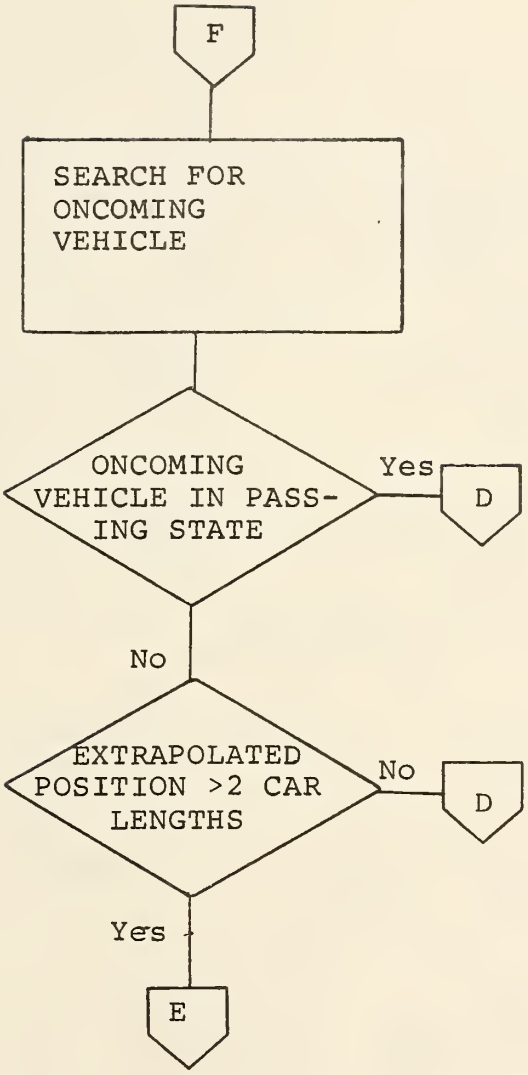


Figure 20. PASPOS Program Flow Chart (Continued)

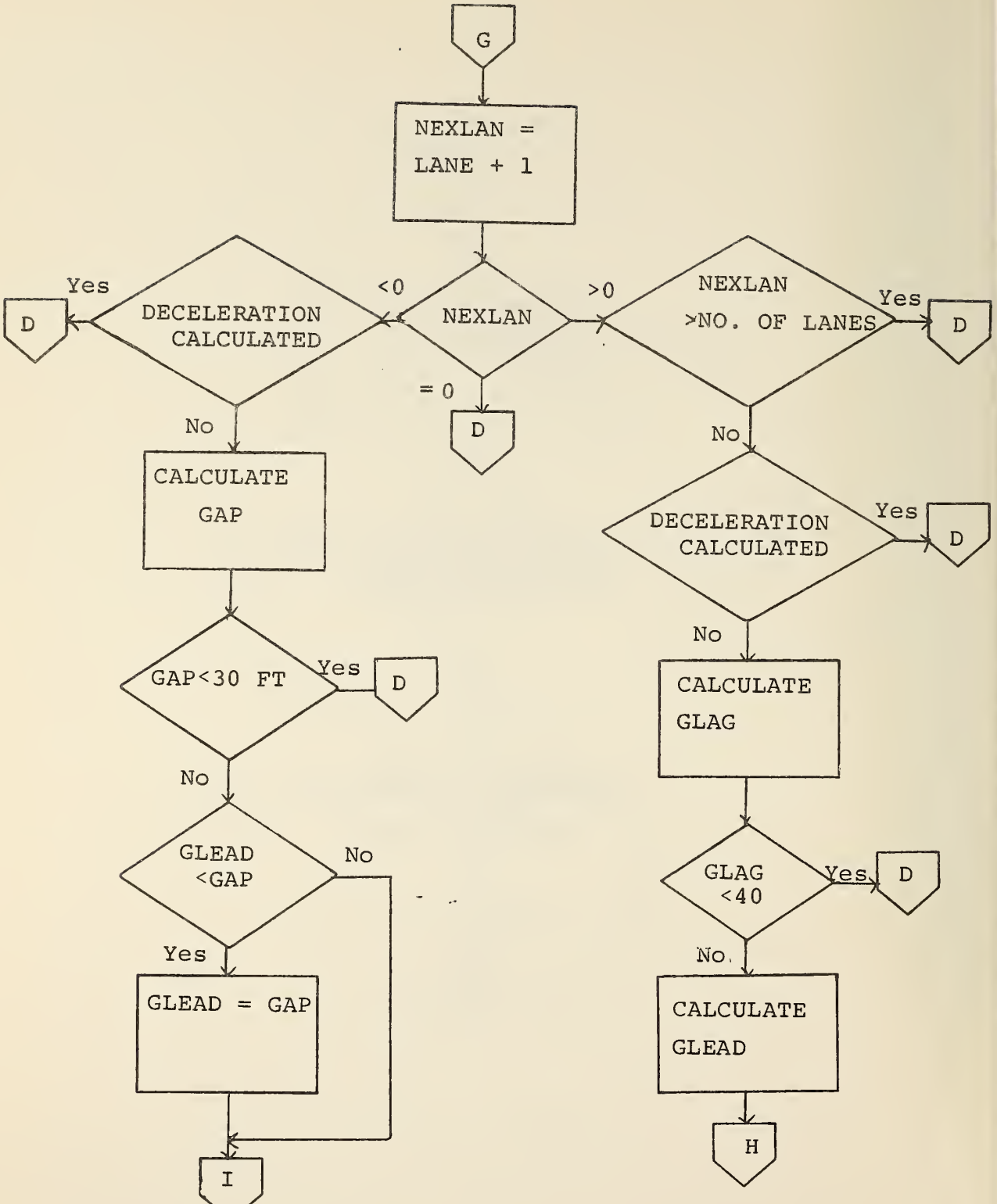


Figure 20. PASPOS Program Flow Chart (Continued)

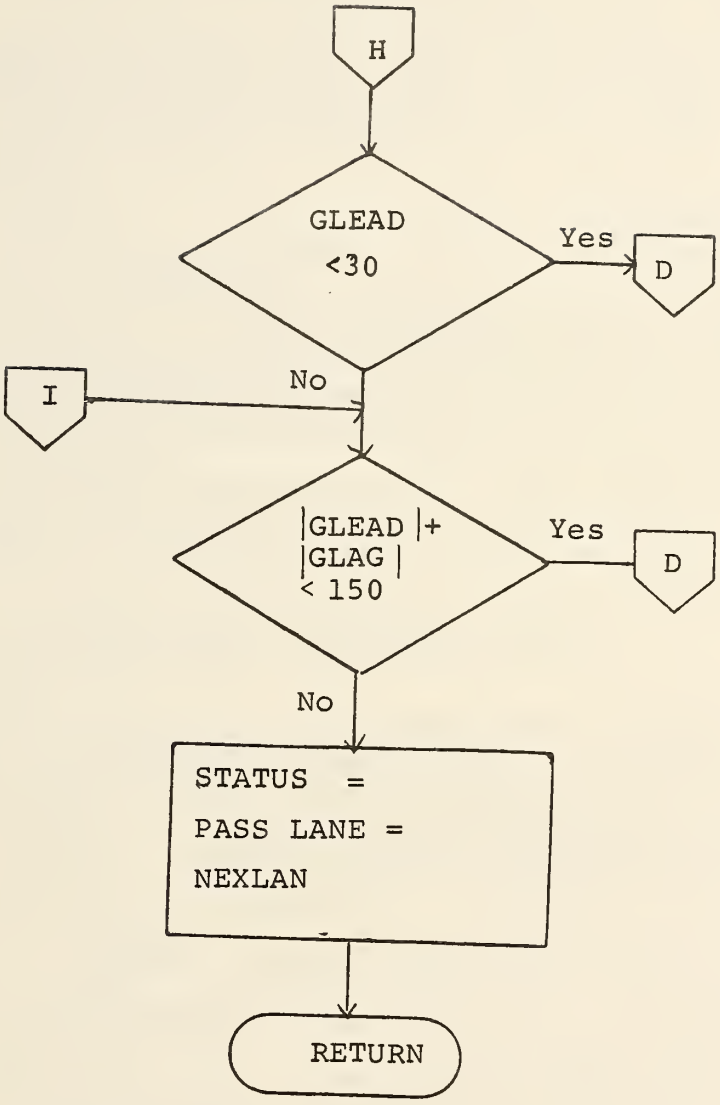


Figure 20. PASPOS Program Flow Chart (Continued)

```
001               SUBROUTINE PASPOS (MUM4,MUP2, MUP3)
                C     CALCULATES WHETHER PASSING IS POSSIBLE USING CONDITIONS: 1) LEAD  00
                C     CAR IS NOT IN PASSING STATE, 2) SUFFICIENT TIME IS AVAILABLE FOR  00
                C     MTH CAR TO PASS LEAD CAR BY ITS OWN VEHICLE LENGTH PLUS 5 FEET    00
                C     WITHOUT DANGER OF COLLISION WITH ONCOMING CAR.                    00
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                  1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                  2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                  3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                  4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                 1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                  3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
                C   BRIDGE LOADING DATA
006               COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                  1     XPOS(50), DXPOS(50), ACCLR(50),
                  2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
                C
007               DIMENSION JFWD(200),JBAK(200),JNDX(200)
008               EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
                 1            (JNDX(1),INDX(201))
                C
009               CALL CALACC                                                       00
010               IF (LANE(MU).GT.0.AND.ACC(MU).LE.0.) GO TO 10
011               IF (LANE(MU).LT.0.AND.ACC(MU).GE.0.) GO TO 10
                C     LEAD CAR MUST BE IN NORMAL STATE                            00
012               IF (ND.EQ.1) GO TO 200
013               IF (MZ.NE.0.AND.Z(MZ).EQ.0.) GO TO 10
014               IF (KSTAT(ILV).NE.0) GO TO 10                                   00
                C VEHICLE BEHIND SHOULD NOT BE ATTEMPTING PASS                    00
015               IF(MUP2.EQ.0)  GO TO 5
016               IF (LANE(MU).EQ.2.OR.LANE(MU).EQ.-NL)  GO TO 5
017               IF (KSTAT(MUP2).GT.0) GO TO 10
018              5 CONTINUE
019               IF(MUM4.EQ.0)  GO TO 91
020               FRTGAP=POS(MUM4) - POS(ILV)
                C ADEQUATE SPACE IN FRONT OF LEAD CAR.                            00
                C FRONT GAP MUST EQUAL 3 CAR LENGTHS.                             00!
021               IF (ABS(FRTGAP).LT.3.*VEHLEN(ITY)) GO TO 10                     00
                C     CALCULATE TIME REQUIRED FOR PASSING                         00!
022               SPDIFF = ABS(SPD(MU,1) - SPD(ILV,1))                            00
```

```
      C     ACCELERATION MUST BE PROVIDE, EITHER AS A CONSTANT, A FUNCTION     00
      C     OF SPEED, OR A FUNCTION OF VEHICLE TYPE.                           00
023           AM=ABS(ACC(MU))
024           C=-SPCK-ABS(FRTGAP)
025           DISCR=SPDIFF*SPDIFF-2.*AM*C                                        00
026           DISCR = ABS(DISCR)
027           IF (ABS(AM).LT.0.0001)  GO TO 10
028           TIMREQ=(SQRT(DISCR)-SPDIFF)/AM                                     00
029           IF (TIMREQ.LT.0) GO TO 900                                         00
      C     CALCULATE MAXIMUM SPEED REQUIRED                                   00
030           SPDABS=ABS(SPD(MU,1))                                              00
031           SPDREQ=SPDABS+AM*TIMREQ                                            00
032           IF(SPDREQ.LE.SPDMAX) GO TO 40                                      00
      C     RECALCULATE TIME REQUIRED                                          00
033           D=SPDMAX-SPDABS                                                    00
034           T1=D/AM                                                            00
035           TSQ=T1*T1                                                          00
036           E=SPDMAX-ABS(SPD(ILV,1))                                           00
037           F=D*T1-0.5*AM*TSQ-C                                                00
038           IF (ABS( E).LT.0.0001)  GO TO 10
039           TIMREQ=F/E                                                         00
      C IF TIME REQUIRED IS TOO GREAT, FOLLOW.                                 00
040           TIMREQ = TIMREQ + 3.0
041           IF (TIMREQ.GT.30.) GO TO 10                                        00
      C     TWO BRANCHES TO DETERMINE SEPARATELY POSSIBILITY OF PASSING FOR    00
      C     ODD OR EVEN M.                                                     00
042        40 IF(LANE(MU).LT.0) GO TO 300                                        00
043           IF (ND.EQ.1)  GO TO 91
      C     SEARCH FOR ONCOMING VEHICLE                                        00
044           JFD = JFWD(1)
045        21 K = JNDX(JFD)
046           IF (POS(K)   - POS(MU))   20,10,30
047        20 IF(JFD.EQ.JBAK(1))  GO TO 90
048           JFD = JFWD(JFD)
049           GO TO 21
      C ONCOMING VEHICLE MUST NOT BE IN PASSING STATE                          00
050        30 IF (KSTAT(K).NE.0) GO TO 10                                        00
      C EXTRAPOLATED POSITION OF ONCOMING VEHICLE MUST DIFFER BY 2 CAR LENGTHS00
      C FROM EXTRAPOLATED POSITION OF VEHICLE.                                 00
051           XC=POS(K)  +SPD(K,1)*TIMREQ-2.*VEHLEN(ITY)
052           XA=POS(MU)  +SPD(MU,1)*TIMREQ+TIMREQ*TIMREQ*ACC(MU)
053           IF (XC.LE.XA) GO TO 10                                             00
054           GO TO 90                                                           00
055        300 IFD = IFWD(1)
056        301 K = INDX(IFD)
057           IF (POS(K)   - POS(MU))   330,10,320
058        320 IF (IFD.EQ.IBAK(1))  GO TO 90
059           IFD = IFWD(IFD)
060           GO TO 301
      C SEE PRECEEDING COMMENTS.                                               00
061        330 IF (KSTAT(K).NE.0) GO TO 10                                       00
062           XC=POS(K) + SPD(K,1)*TIMREQ+2.*VEHLEN(ITY)
063           XA=POS(MU) + SPD(MU,1)*TIMREQ+TIMREQ*TIMREQ*ACC(MU)
064           IF (XC.GE.XA) GO TO 10                                             00
```

```
065           91 CONTINUE
066          90   KSTAT(MU) = 1                                                    00
067               IF (LANE(MU).GT.0)  LANE(MU) = 2
068               IF (LANE(MU).LT.0)  LANE(MU) = -1
069               GO TO 100                                                       00
070          10   KSTAT(MU) = -1                                                  00
071               GO TO 100                                                       00
072           200 NEXLAN=LANE(MU)+1                                               00
073               IF (NEXLAN) 210,10,220                                          00
074           210 IF (ACC(MU).GT.0) GO TO 10
075               GAP = POS(MUP3)-POS(MU)
            C FOR OPPOSING VEHICLES,GLEAD IS NEGATIVE,GLAG POSTIVE.               00
076           240 IF (-GAP.LT.30.) GO TO 10                                       00
077               IF (GLEAD.LT.GAP) GLEAD=GAP                                     00
078               GO TO 600                                                       00
            C
079           220 IF (NEXLAN.GT.NL) GO TO 10                                      00
080               IF (ACC(MU).LT.0) GO TO 10
081               GLAG=-BRPOS                                                     00
082               GLEAD=BRPOS                                                     00
            C FOR FORWARD VEHICLES,GLEAD IS POSITIVE, GLAG NEGATIVE.              00
083               IF (MUP3.EQ.0)  GO TO 280
084               GLAG = POS(MUP3)-POS(MU)
085               IF (-GLAG.LT.40)  GO TO 10
086           280 IF(ILV.EQ.0)  GO TO 282
087               IF (LANE(ILV).EQ.NEXLAN)  GLEAD = POS(ILV) - POS(MU)
088               IF(MUM4.EQ.0)  GO TO 282
089               IF (LANE(MUM4).EQ.NEXLAN)  GLEAD = POS(MUM4) - POS(MU)
090           282 IF (GLEAD.LT.30)  GO TO 10
091           600 IF (ABS(GLEAD)+ABS(GLAG).LT.150.) GO TO 10                      00
092               KSTAT(MU) = 1
093               LANE(MU)=NEXLAN                                                 00
094               GO TO 100                                                       00
095           900 WRITE (6,910) TIMREQ                                            00
096           910 FORMAT (1H 'REQUIRED PASSING TIME IS NEGATIVE', F10.3)          00
097           100 CONTINUE
098               RETURN
099                END                                                            00
```

PASTES

The PASTES subroutine is called by UPDATE to determine whether a passing vehicle has completed its maneuver and may return to the right lane. On a two lane, two way road, a maneuvering vehicle will resume lane as soon as possible. The criteria for this are:

$$x_M = x_L + H_M = 2D \text{ and} \tag{9}$$

$$x_M = x_F - H_F - 2D$$

This is equivalent to an opening equal to the length of the maneuvering vehicle plus 40 feet. If such an opening does not exist and the maneuvering vehicle must pass two vehicles, it will resume lane when

$$x_M = x_F + H_M + 2D \tag{10}$$

with no further check.

The maneuvering vehicle on a multilane highway will move to the right when

$$x_M = x_B + H_M + 6D \tag{11}$$

that is, the nearest vehicle in the right lane is at least 60 feet behind and the nearest vehicle ahead leads by a desired spacing.

If it is determined that a pass is complete, an acceleration is calculated that will reduce the vehicle speed to its generated speed, the lane is changed to the right lane and the status is set to free running, that is, 0.

No subroutines are called by PASTES.

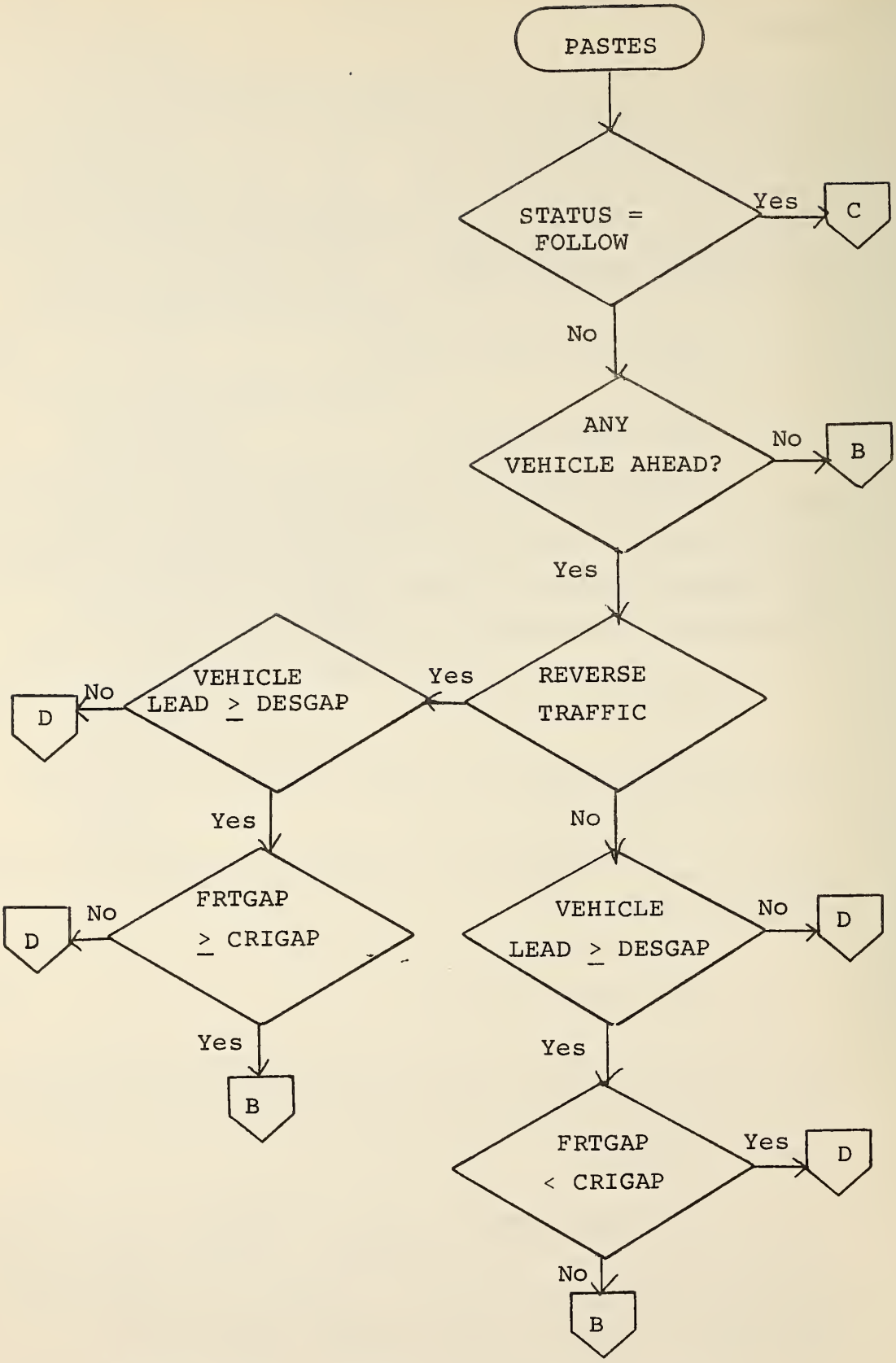


Figure 21. PASTES Program Flow Chart

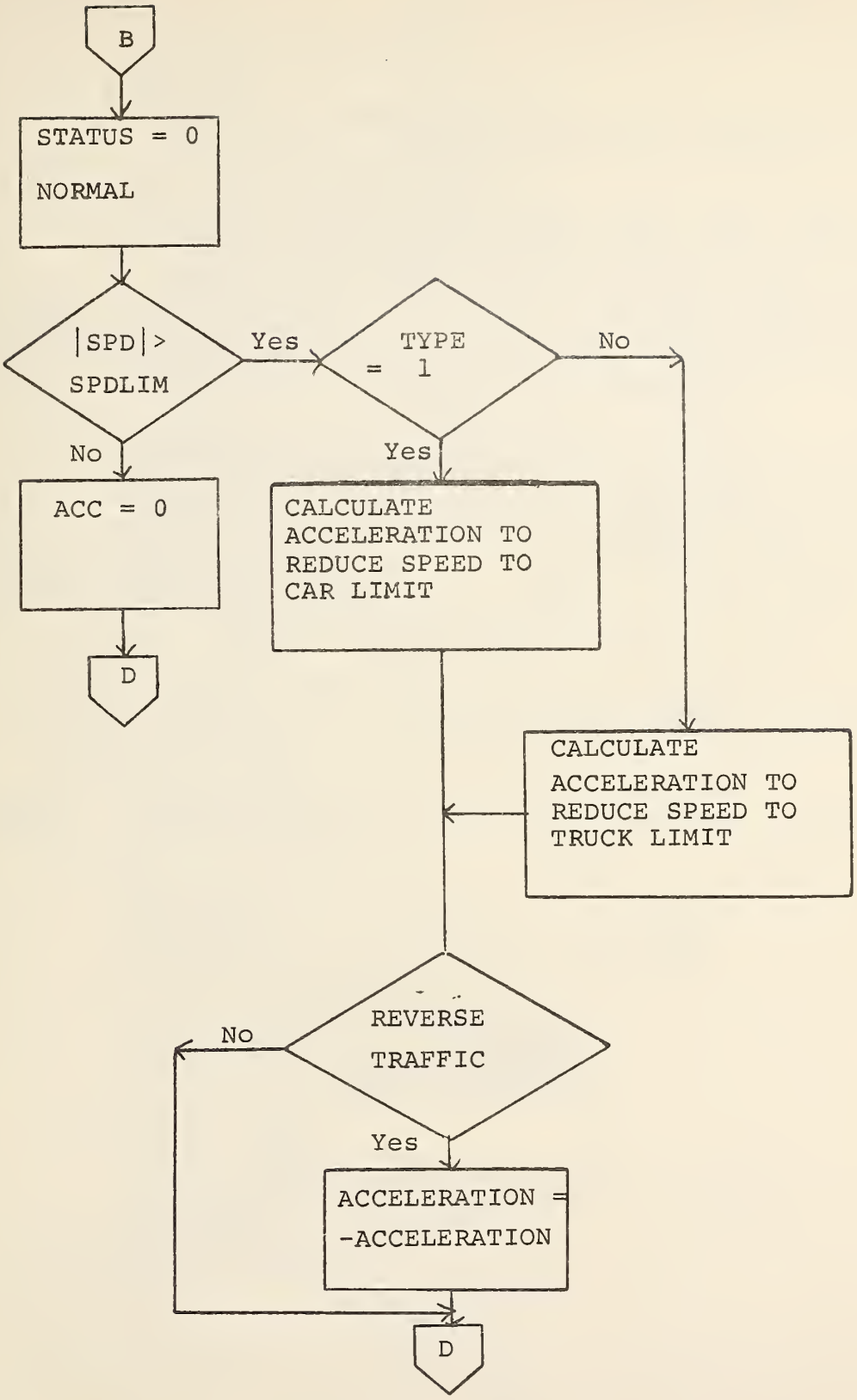


Figure 21. PASTES Program Flow Chart (Continued)

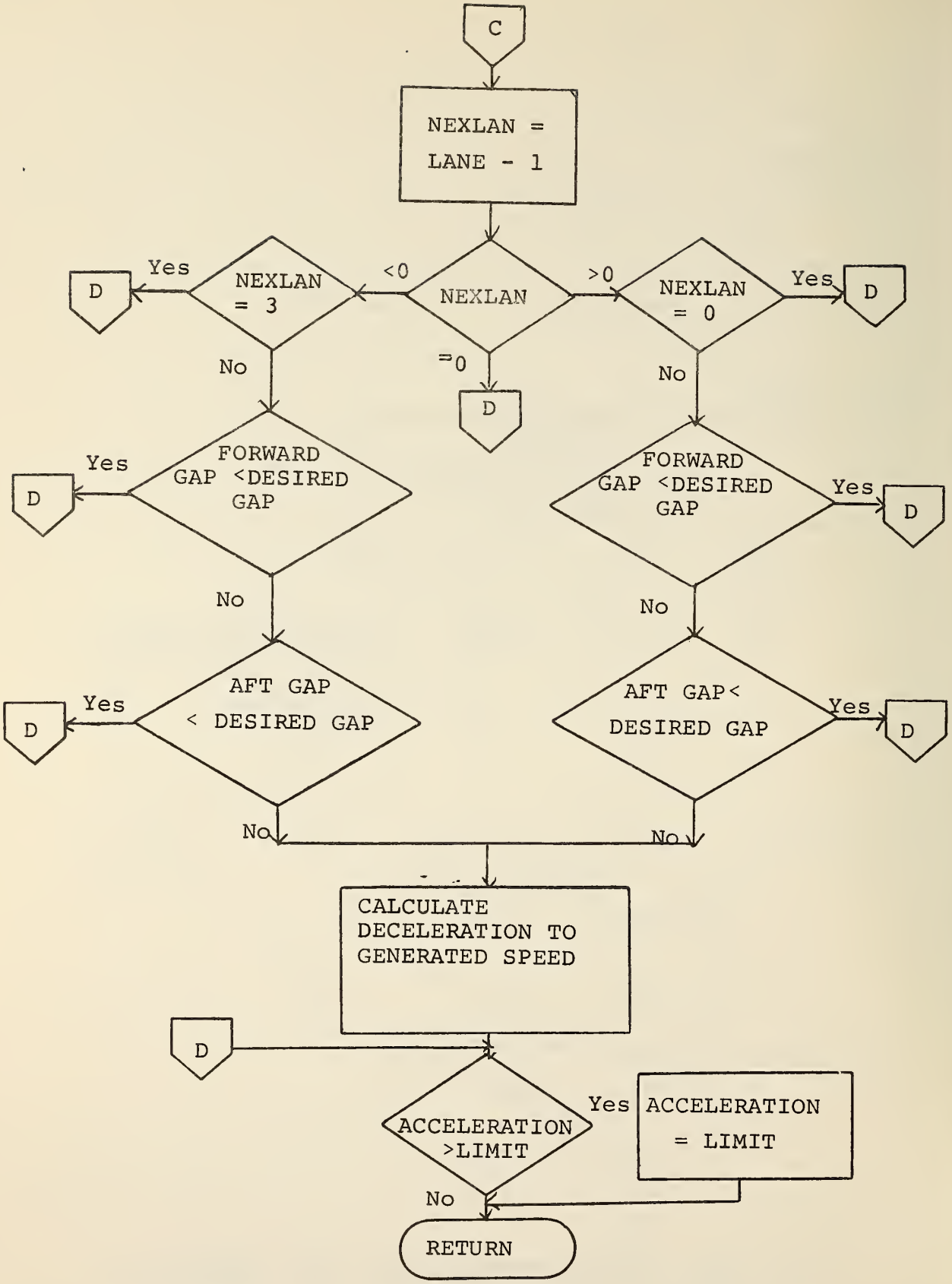


Figure 21. PASTES Program Flow Chart (Continued)

```
001             SUBROUTINE PASTES(MUP2)
                C     DETERMINE IF PASS IS COMPLETED.                                    00
                C   VEHICLE DATA
002             COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003             COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004             COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005             COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
                C   BRIDGE LOADING DATA
006             COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                1     XPOS(50), DXPOS(50), ACCLR(50),
                2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
                C
                C FOR TWO LANE OPPOSING TRAFFIC,VEHICLE IN PASSING STATE IS ASSUMED    00
                C TO BE IN OPPOSING LANE. WHEN CAN IT RESUME NORMAL STATE - WHEN IT HAS 00
                C PASSED LEAD CAR          AND IF SUFFICIENT SPACE IS AVAILABLE BETWEEN00
                C LEAD VEHICLE AND ITS LEADING VEHICLE.                                00
                C
007             IF (MUP2.EQ.0) GO TO 10
008             IF(KSTAT(MU).LT.1) GO TO 60                                            00
                C
009             IF (ILV.EQ.0) GO TO 15
010             SPCK = POS(ILV) - POS(MU)
011             CRIGAP=VEHLEN(ITY)+2.*SAFDIS                                           00
012             L=ITYPE(ILV)
013             DESGAP = ABS(GAPFAC*SPD(MU,1)) + VEHLEN(L)
014             IF (LANE(MU).LT.0) GO TO 70                                            00
015             IF (SPCK.LE.DESGAP) GO TO 10
016             FRTGAP=POS(ILV)  - POS(MUP2)
017             CRIGAP=CRIGAP+VEHLEN(L)+SAFDIS*2.                                      00
018             IF(FRTGAP.LT.CRIGAP) GO TO 10                                          00
019             GO TO 15                                                               00
020           70  CRIGAP=-CRIGAP                                                      00
021             DESGAP = -DESGAP
022             IF (SPCK.GE.DESGAP) GO TO 10
023             FRTGAP=POS(ILV)  - POS(MUP2)
024             CRIGAP=CRIGAP-VEHLEN(L)-SAFDIS*2.                                      00
025             IF (FRTGAP.GT.CRIGAP) GO TO 10                                         00
026             GO TO 15                                                               00
```

```
027          15 KSTAT(MU) = 0
028          35 IF (ABS(SPD(MU,1)).GT.SPDLIM) GO TO 30
029             ACC(MU) = 0.0
030             GO TO 10                                                    00
031          30 IF (ITY.NE.1) GO TO 40                                      00
032             ACC(MU) = (SPDLIM -ABS(SPD(MU,1)))/DELTIM
033             GO TO 50                                                    00
034          40 CONTINUE
035             ACC(MU) = (TRKLIM -ABS(SPD(MU,1)))/DELTIM
036          50 IF(LANE(MU).LT.0)  ACC(MU) = -ACC(MU)
037             GO TO 10                                                    00
        C
038             60 NEXLAN=LANE(MU)-1                                        00
039             GLAG=VEHLEN(ITY)+6.*SAFDIS                                  00
040             IF(NEXLAN)200,10,250                                        00
041          200 CONTINUE
042             IF (NEXLAN.EQ.-3)  GO TO 10
043             IF (ILV.EQ.0) GO TO 240
044             GAP=POS(MU) - POS(ILV)
045             IT=ITYPE(ILV)
046             GLEAD=VEHLEN(IT)+3.*SAFDIS                                  00
047             IF (GAP.LT.GLEAD) GO TO 10                                  00
048          240 CONTINUE
049             GAP = POS(MU) - POS(MUP2)
050             IF(ABS(GAP).LT.GLAG) GO TO 10
051             GO TO 290                                                   00
        C   OLD LANE WAS = 2
052          250 CONTINUE
053             IF (NEXLAN.EQ.0) GO TO 10
054             IF (ILV.EQ.0) GO TO 255
055             GAP=POS(MU) - POS(ILV)
056             IT=ITYPE(ILV)
057             GLEAD=VEHLEN(IT)+3.*SAFDIS                                  00
058             IF (ABS(GAP).LT.GLEAD) GO TO 10                             00
059          255 CONTINUE
060             GAP = POS(MU) - POS(MUP2)
061          280 IF(GAP.LT.GLAG) GO TO 10                                   00
062          260 CONTINUE                                                   00
063          290 ACC(MU) = (SPD(MU,2) - SPD(MU,1))/DELTIM
064             LANE(MU)=NEXLAN                                             00
065           10 CONTINUE
066             IF (ABS(ACC(MU)).GT.ACCEL)  ACC(MU) = SIGN(ACCEL,ACC(MU))
067             RETURN
068             END                                                         00
```

RANF

RANF is a random number function generator. This routine was copied from the IBM/SSP subroutine RANDU.

No subroutines are called by RANF.

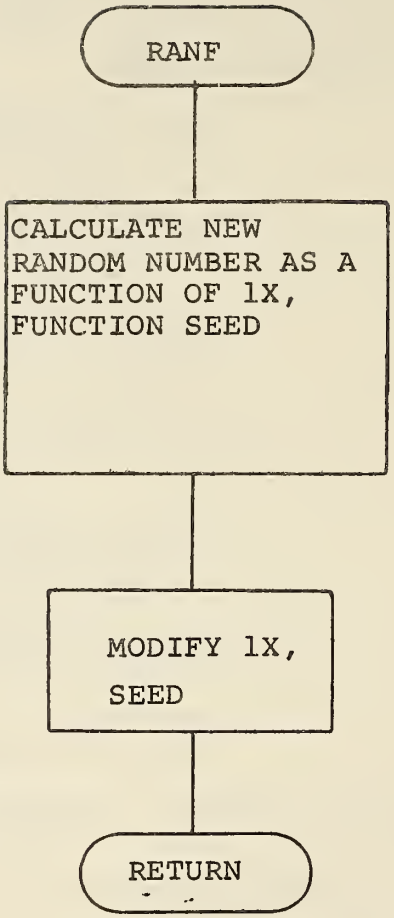


Figure 22. RANF Program Flow Chart

```
001              FUNCTION RANF(DUMMY)                                          OC
                 C                                                             00
                 C     THIS FUNCTION IS BASICALLY SUBROUTINE RANDU FROM THE IBM/SSP.   OC
                 C     ITS NAME AND ARGUMENT LIST HAVE BEEN CHANGED IN ORDER THAT NO   00
                 C     CHANGES WOULD BE NECESSARY IN THE CALLING PROGRAM.      OC
                 C     THE RETURNED VALUE OF RANF IS IN THE RANGE 0. TO 1.     00
                 C                                                             OC
002              COMMON /RANDOM/ IX,IY,YFL                                     00
003              IY = IX * 65539                                               OC
004              IF (IY) 5,6,6                                                 00
005            5 IY = IY + 2147483647+1                                        OC
006            6 YFL = IY                                                      00
007              IX = IY                                                       OC
008              YFL = YFL * .4656613E-9                                       00
009              RANF = YFL                                                    OC
010              RETURN                                                        00
011              END                                                           OC
```

READ

The subroutine READ reads any or all of the following subperiod data:

Vehicle type distribution - by direction
Vehicle headway distribution - by direction
Vehicle speed distribution - by type
Vehicle weight distribution - by type
Platoon distribution - by direction

Any data which is read in is printed out. If either the Traffic Distribution or Platoon Distribution tables are changed, the Traffic Distribution is modified so that the generation of platoons does not increase the number of trucks generated. The unmodified Traffic Distribution is printed out.

No subroutines are called by READ.

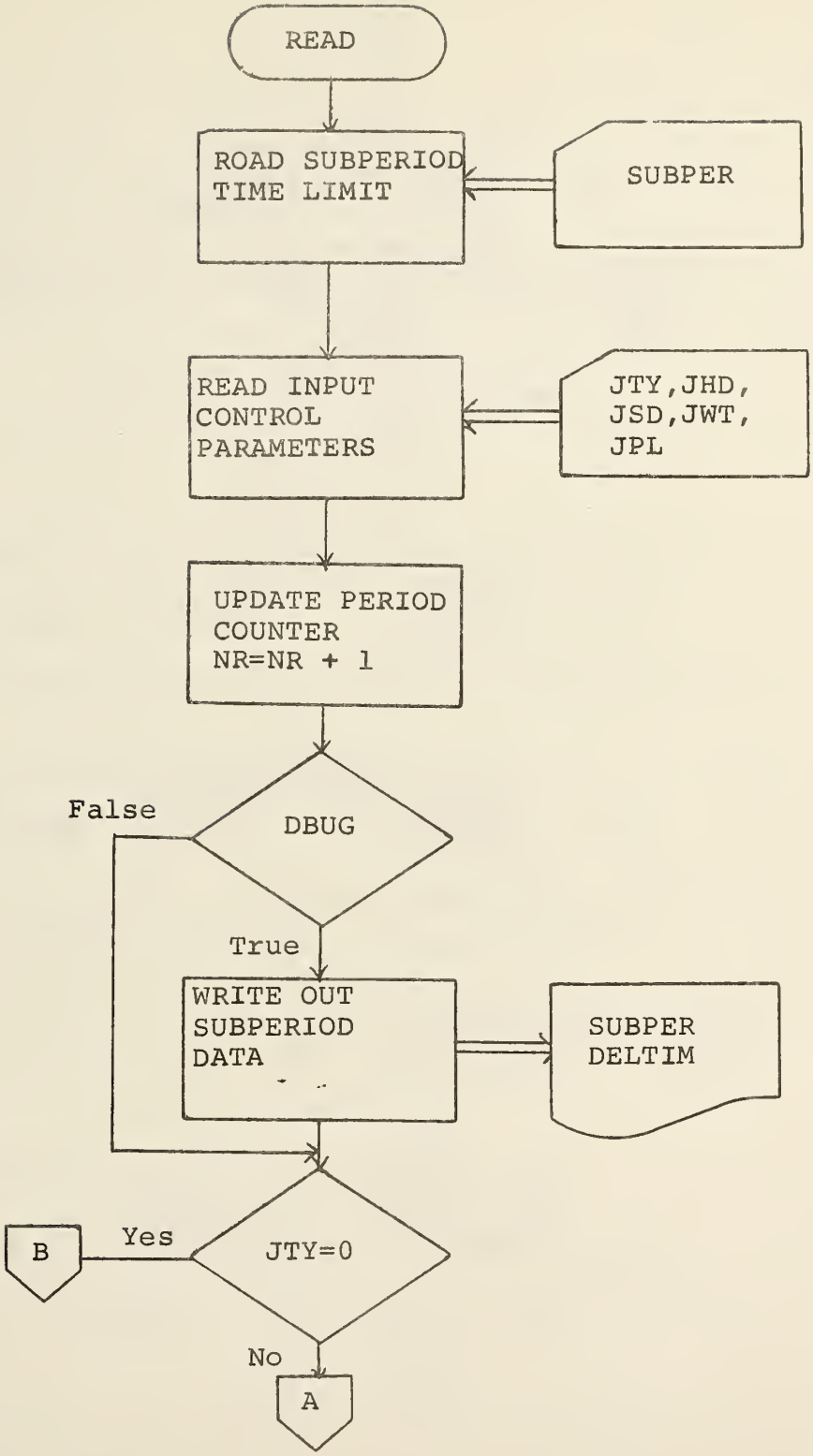


Figure 23. READ Program Flow Chart

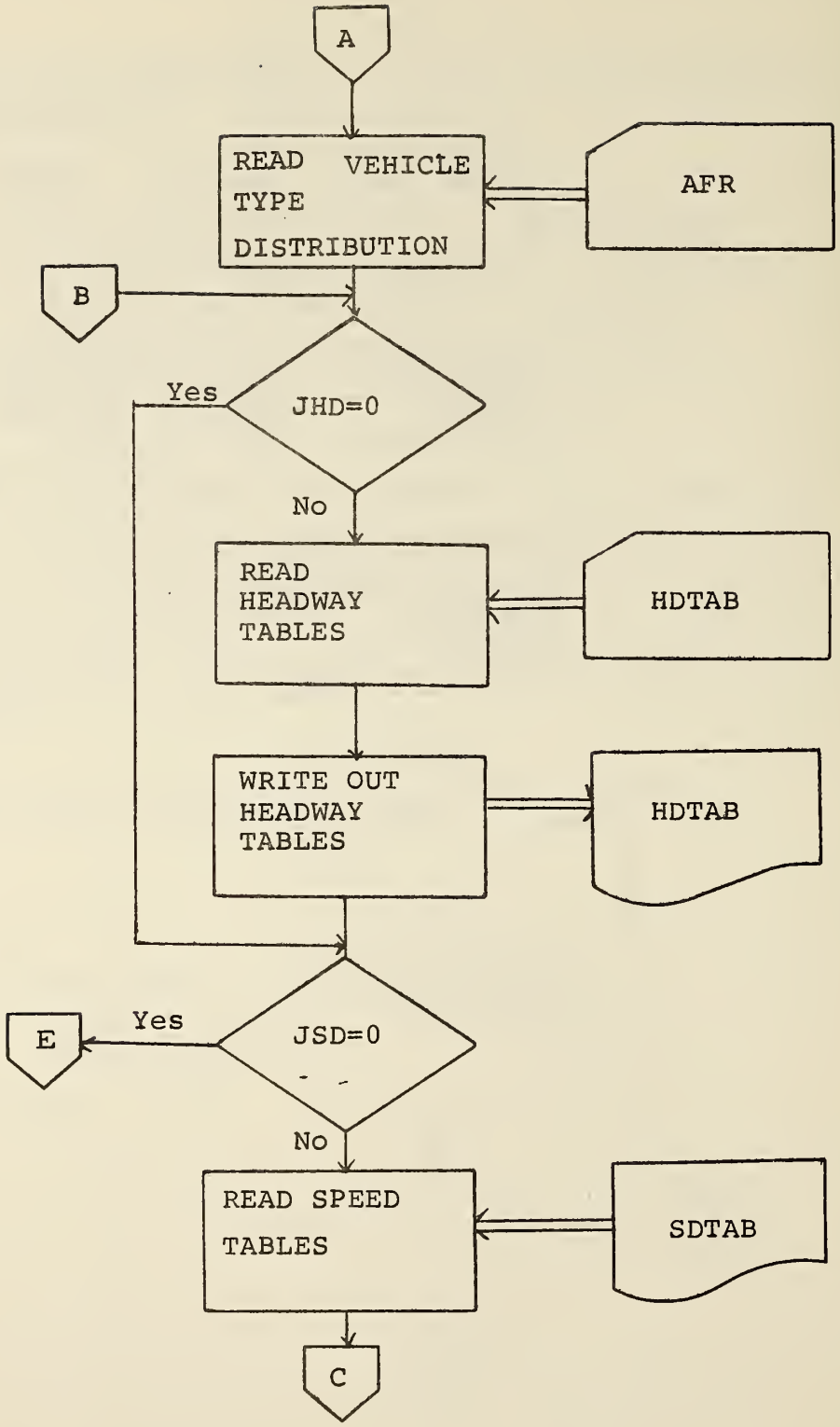


Figure 23. READ Program Flow Chart (Continued)

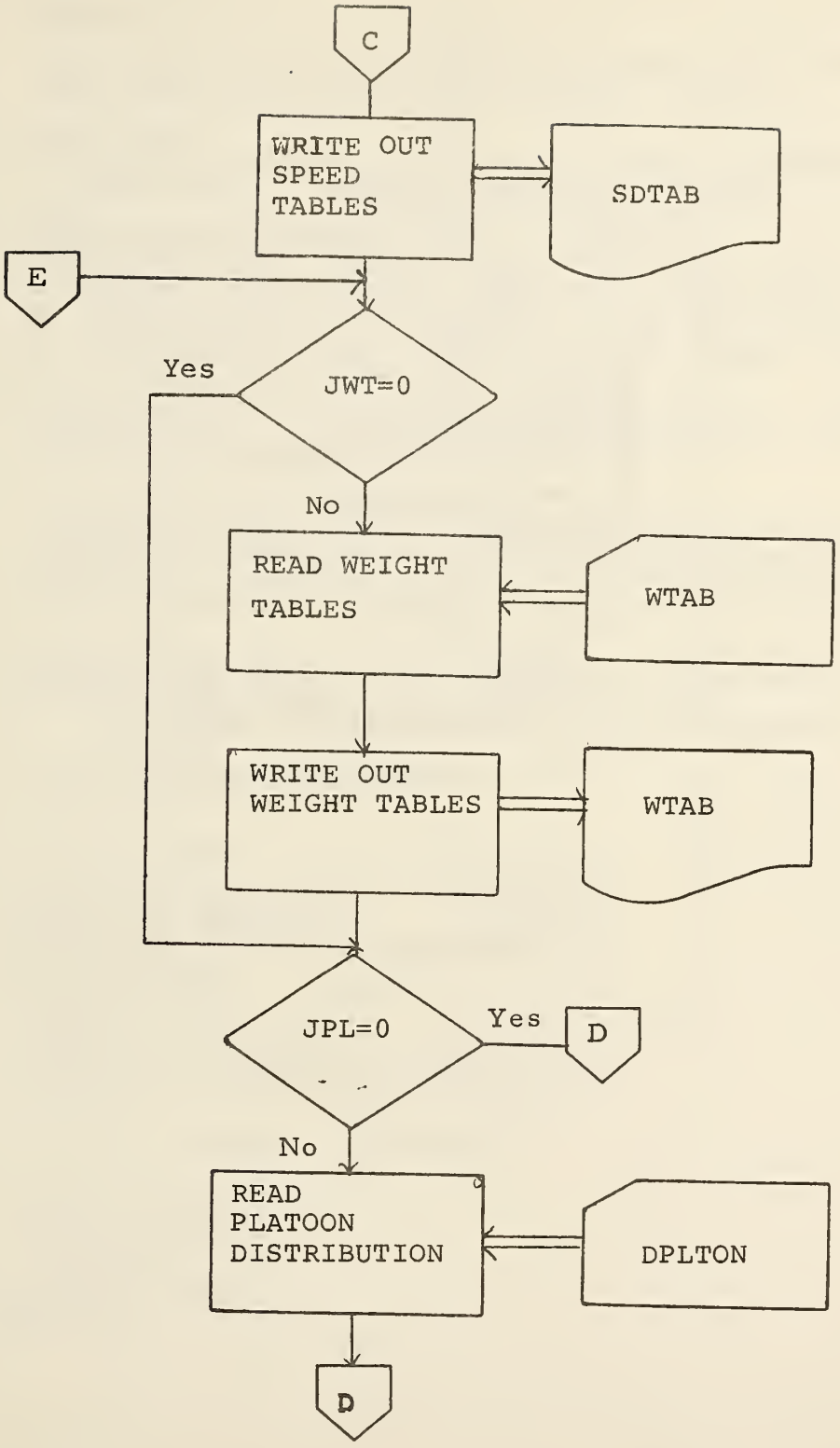


Figure 23. READ Program Flow Chart (Continued)

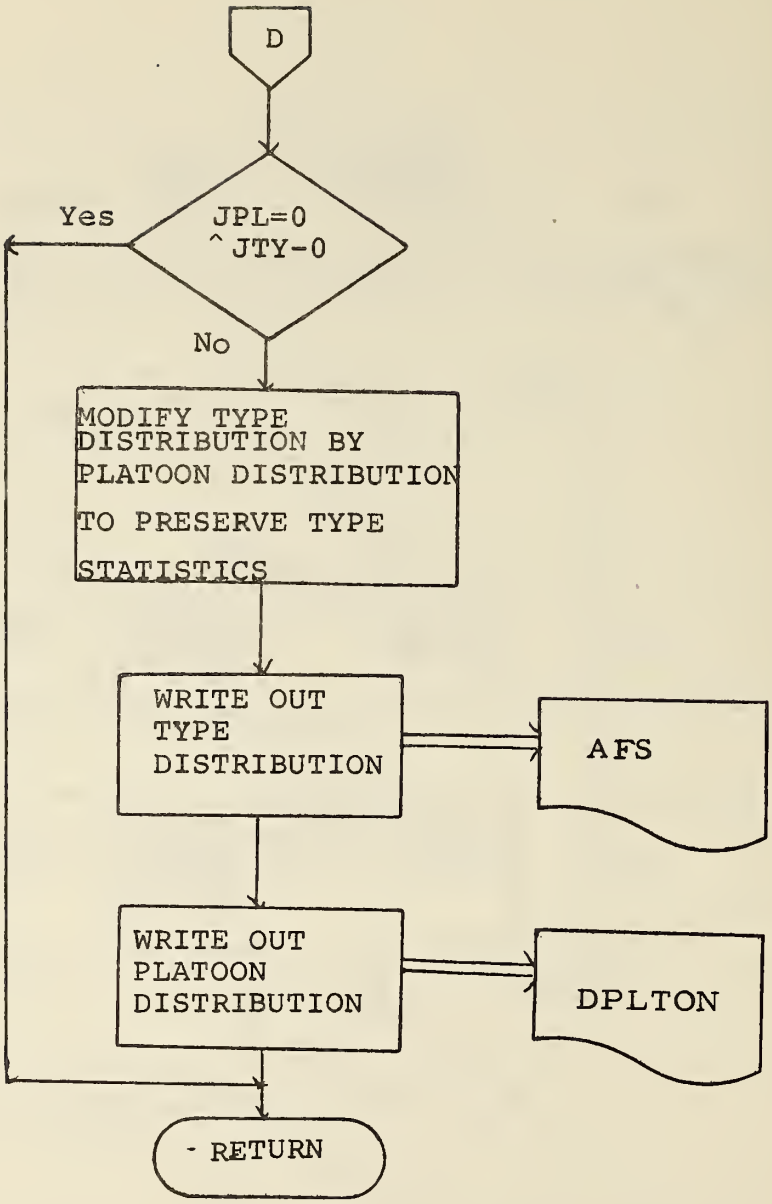


Figure 23. READ Program Flow Chart (Continued)

```
001             SUBROUTINE READ                                                 00
                C
                C     READS CONTROL PARAMETER FOR EACH SUBPERIOD. READS TABLES.  00
                C     PRINTS OUT TIME OF SUBPERIOD AND FACTOR, IF NFB=1          00
                C
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                  1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                  2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                  3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                  4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                  1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                  1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                  2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                  3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
                C   BRIDGE LOADING DATA
006               COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                  1     XPOS(50), DXPOS(50), ACCLR(50),
                  2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
                C
007               LOGICAL TRUCK, PLATON, DBUG
                C
                C
                C  INITIALIZE UPDATE
                C
008                 READ (5,101) SUBPER
009               101 FORMAT (F6.0)
010                 READ (5,150) JTY,JHD,JSD,JWT,JPL
011               150 FORMAT (20I4)                                             00
012                 NR=NR+1                                                     00
013                 IF (.NOT.DBUG)  GO TO 20
014                 WRITE (6,310) SUBPER,       DELTIM
015               310 FORMAT (1H ,'SUBPERIOD=',E15.2,' SECONDS  ',
                     $'UPDATE PASS EVERY',F10.2)                                00
                C
                C   INITIALIZE FREQUENCY DISTRIBUTION
                C
016                20 DO 30 M=1,10
017                   DO 35 MN=1,LV                                             00
018                35 FREQ(MN,M  )=0.0
019             30    CONTINUE                                                  00
020               110 FORMAT (10F8.2)                                           00
021               100 FORMAT (I4)                                               00
```

```
0022              IF (JTY.EQ.0) GO TO 50                                    00
                C
                C   READ VEHICLE TYPE DISTRIBUTIONS
                C
0023              IF (MD.GT.1)  GO TO 10                                    00
0024              AFS(1,1)=1.0
0025              GO TO 45
0026           10 DO 11 J=1,JTY                                             00
0027           11 READ (5,210) (AFS(M,J),M=1,MD)
0028          210 FORMAT (10F8.4)
0029              DO 42 J=1,JTY
0030              DO 42 M=2,MD
0031              AFS(M,J)=AFS(M-1,J)+AFS(M,J)
0032           42 CONTINUE
0033           45 IF (JTY.EQ.ND) GO TO 50
0034              DO 40 M=1,MD                                              00
0035           40 AFS(M,2)=AFS(M,1)
0036           50 IF(JHD.EQ.0) GO TO 60                                     00
                C
                C   READ HEADWAY TABLES
                C
0037              DO 55 J=1,JHD                                             00
0038              READ(5,100)NUMD                                           00
0039              LHD(J)=NUMD                                               00
0040              READ(5,110)(HDTAB(M,J),M=1,NUMD)                          00
0041           55 DELHD(J)=1.0/(NUMD-1)                                     00
0042              IF(JHD.EQ.ND)GO TO 59
0043              LHD(2)=LHD(1)                                             00
0044              DELHD(2)=DELHD(1)                                         00
0045              DO 56 M=1,NUMD                                            00
0046           56 HDTAB(M,2)=HDTAB(M,1)                                     00
0047           59 CONTINUE
0048              WRITE(6,3000)                                             00
0049              WRITE(6,3000)                                             00
0050              WRITE(6,3008)                                             00
0051              WRITE(6,3009) (I,I=1,20)                                  00
0052              WRITE(6,3004)                                             00
0053              J=1                                                       00
0054              WRITE(6,3010) J,(HDTAB(I,1),I=1,20)                       00
0055              J=2                                                       00
0056              WRITE(6,3010) J,(HDTAB(I,2),I=1,20)                       00
0057              WRITE(6,3000)                                             00
0058              WRITE(6,3009) (I,I=21,40)                                 00
0059              WRITE(6,3004)                                             00
0060              J=1                                                       00
0061              WRITE(6,3010) J,(HDTAB(I,1),I=21,40)                      00
0062              J=2                                                       00
0063              WRITE(6,3010) J,(HDTAB(I,2),I=21,40)                      00
                C
                C   READ SPEED TABLES
                C
0064           60 IF(JSD.EQ.0) GO TO 70                                     00
0065              DO 80 I=1,JSD                                             00
0066              READ(5,100)NUMD                                           00
```

```
067              READ(5,110)(SDTAB(M,I),M=1,NUMD)                          00
068              LSP(I)=NUMD                                               00
069              DELSD(I)=1.0/(NUMD-1)                                     00
070           80 CONTINUE                                                  00
071              IF(JSD.EQ.MD) GO TO 69
072              I=I+1                                                     00
073              DO 85 K=I,MD                                              00
074              DO 86 J=1,NUMD                                            00
075              SDTAB(J,K)=SDTAB(J,JSD)                                   00
076           86 CONTINUE                                                  00
077              LSP(K)=LSP(JSD)                                           00
078              DELSD(K)=DELSD(JSD)                                       00
079           85 CONTINUE                                                  00
080           69 CONTINUE
081              WRITE(6,3000)                                             00
082              WRITE(6,3000)                                             00
083              WRITE(6,3012)                                             00
084              WRITE(6,3003) (I,I=1,20)                                  00
085              WRITE(6,3013)                                             00
086              DO 740 J=1,20                                             00
087              WRITE(6,3014)J,(SDTAB(J,I),I=1,20)                        00
088         740 CONTINUE                                                   00
            C
            C   READ WEIGHT TABLES
            C
089           70 IF (JWT.EQ.0) GO TO 79
090              DO 75 M=1,JWT                                             00
091              READ (5,100) NUMD                                         00
092              LWT(M)=NUMD                                               00
093              DELWT(M)=1.0/(NUMD-1)                                     00
094           75 READ(5,110) (WTAB(N,M),N=1,NUMD)                          00
095              WRITE(6,3000)                                             00
096              WRITE(6,3000)
097              WRITE(6,3011)                                             00
098              WRITE(6,3030) (I,I=1,12)                                  00
099              WRITE(6,3013)                                             00
100              DO 730 J=1,30                                             00
101              WRITE(6,3414) J,(WTAB(J,I),I=1,12)                        00
102         730 CONTINUE                                                   00
103              IF(MD.LT.13)  GO TO 736                                   00
104              WRITE(6,3001)                                             00
105              WRITE(6,3011)                                             00
106              WRITE(6,3030) (I,I=13,20)                                 00
107              WRITE(6,3013)                                             00
108              DO 735 J=1,50                                             00
109              WRITE(6,3414) J,(WTAB(J,I),I=13,20)                       00
110         735 CONTINUE                                                   00
111         736 CONTINUE                                                   00
112           79 CONTINUE
113              IF(JPL.EQ.0)  GO TO 89
            C
            C   READ PLATOON DISTRIBUTIONS
            C
114              DO  88 J=1,JPL
```

```
115              READ(5,100)NUMD
116              READ (5,110) (DPLTON(M,J),M=1,NUMD)
117           88 CONTINUE
118              IF(JPL.EQ.ND)GO TO 87
119              DO 187 M=1,NUMD
120          187 DPLTON(M,2) = DPLTON(M,1)
121           87 CONTINUE
          C
122           89 CONTINUE
123              IF (JPL.EQ.0.AND.JTY.EQ.0) GO TO 1000
124              DO 90 I=1,20
125              DO 90 J=1,ND
126              AFR(I,J)= AFS(I,J)+(1.0-AFS(I,J))*(1-DPLTON(1,J))
127           90 CONTINUE
          C
128              WRITE(6,3000)                                          00(
129              WRITE(6,3002)                                          00(
130              WRITE(6,3000)                                          00(
131              WRITE(6,3003) (I,I=1,20)                               00(
132              WRITE(6,3004)                                          00(
133              DO 710 J=1,2                                           00(
134              WRITE(6,3005) J,(AFS(I,J),I=1,20)
135            710 CONTINUE                                             00(
136              WRITE(6,3000)                                          00(
137              WRITE(6,3000)                                          00(
138              WRITE(6,3000)                                          00(
139              WRITE(6,3006)                                          00(
140              WRITE(6,3000)                                          00(
141              WRITE(6,3070) (I,I=1,10)                               00(
142              WRITE(6,3004)                                          00(
143              DO 720 J=1,2                                           00(
144              WRITE(6,3050) J,(DPLTON(I,J),I=1,10)                   00(
145            720 CONTINUE                                             00(
146           3000 FORMAT(1H0)                                          00(
147           3001 FORMAT(1H1)                                          00(
148           3002 FORMAT(59X,20HTRAFFIC DISTRIBUTION)                  00(
149           3003 FORMAT(2X,12HVEHICLE TYPE,3X,I2,19(4X,I2))           00(
150           3004 FORMAT(4X,9HDIRECTION)                               00(
151           3005 FORMAT(7X,I2,5X,F5.3,19(1X,F5.3))                    00(
152           3006 FORMAT(55X,26HTRUCK PLATOON DISTRIBUTION)            00(
153           3008 FORMAT(61X,15HHEADWAY  TABLES)                       00(
154           3009 FORMAT(2X,12HVALUE NUMBER,3X,I2,19(4X,I2))           00(
155           3010 FORMAT(7X,I2,5X,F5.2,19(1X,F5.2))                    00(
156           3012 FORMAT(59X,14HSPEED   TABLES)                        00(
157           3011 FORMAT(59X,14HWEIGHT  TABLES)                        00(
158           3013 FORMAT(6X,5HVALUE)                                   00(
159           3014 FORMAT(7X,I2,5X,F5.0,19(1X,F5.0))                    00(
160           3030 FORMAT(2X,12HVEHICLE TYPE,3X,I2,11(6X,I2))           00(
161           3050 FORMAT(7X,I2,5X,F5.3, 9(2X,F6.3))                    00(
162           3070 FORMAT(1X,16HNUMBER OF TRUCKS,I2, 9(6X,I2))          00(
163           3414 FORMAT(7X,I2,3X,F7.0,11(1X,F7.0))                    00(
164           1000 RETURN                                               00(
165                END                                                  00(
```

REZONE

This subroutine determines whether or not a vehicle is in a restricted zone. If the vehicle is in a restricted zone, MZ is set to the appropriate zone, positive for upgrade and negative for downgrade.

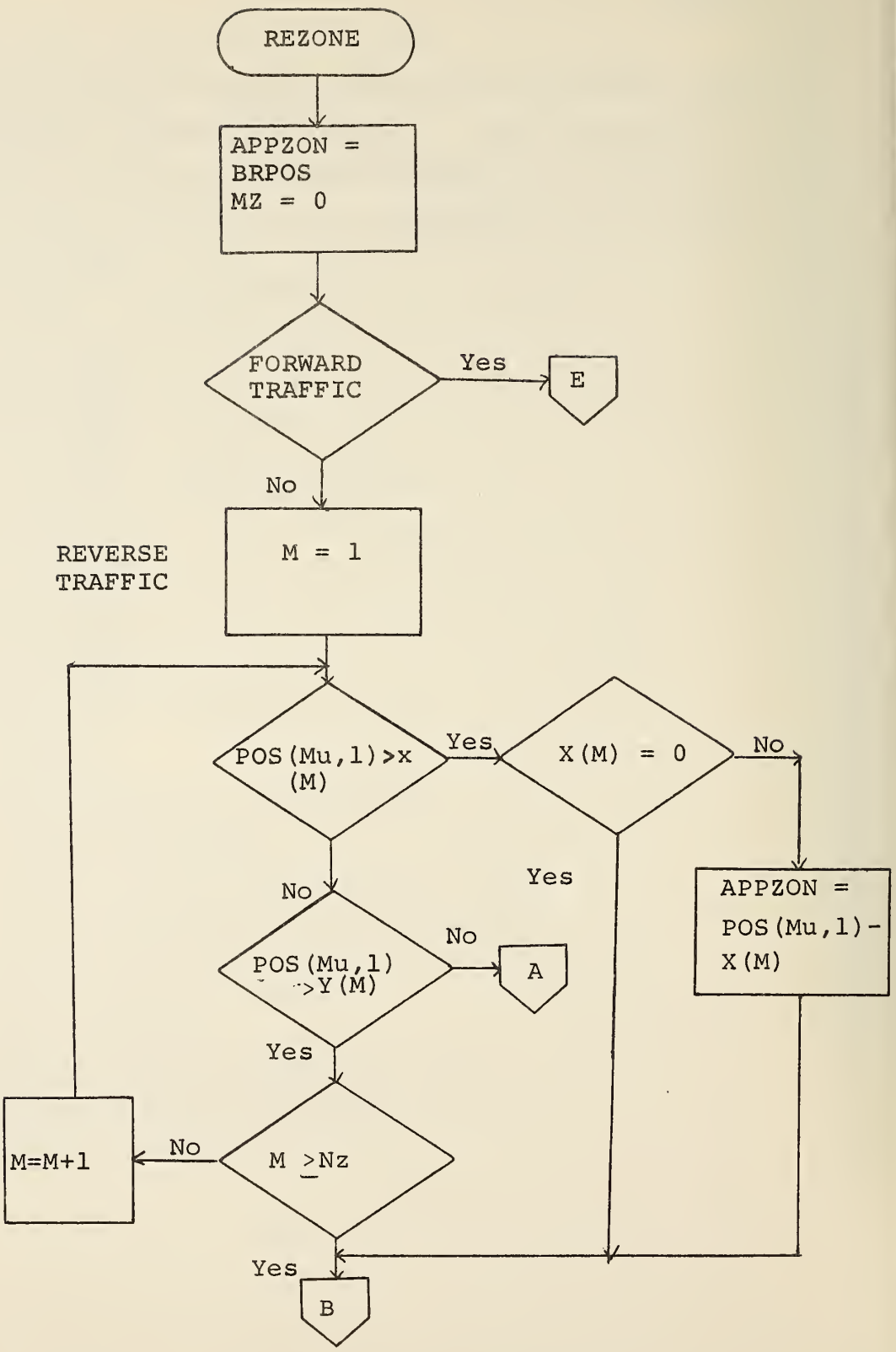


Figure 24. REZONE Program Flow Chart

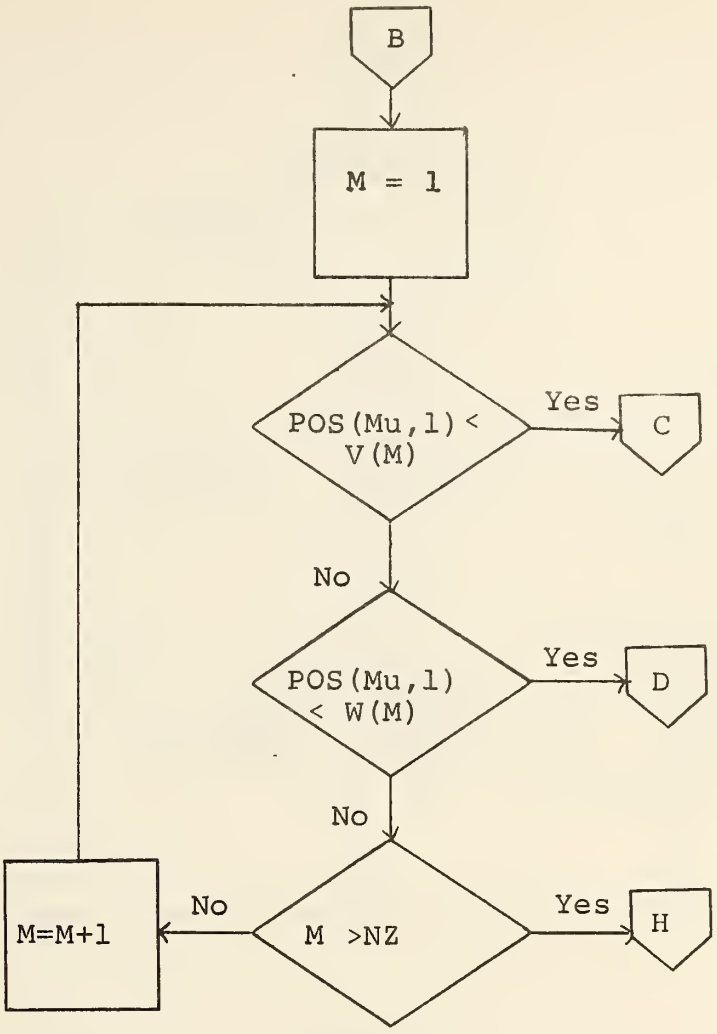


Figure 24. REZONE Program Flow Chart (Continued)

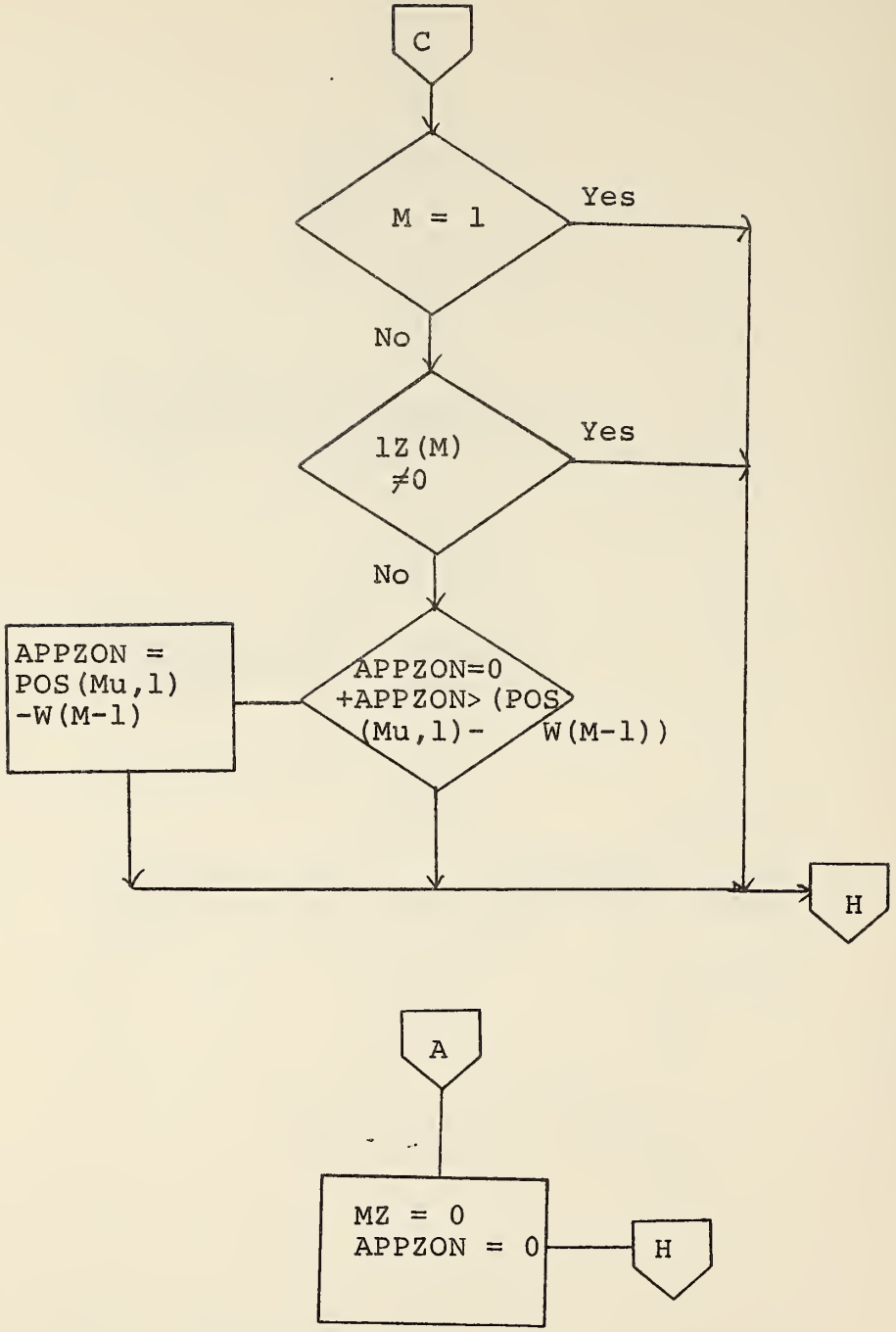


Figure 24. REZONE Program Flow Chart (Continued)

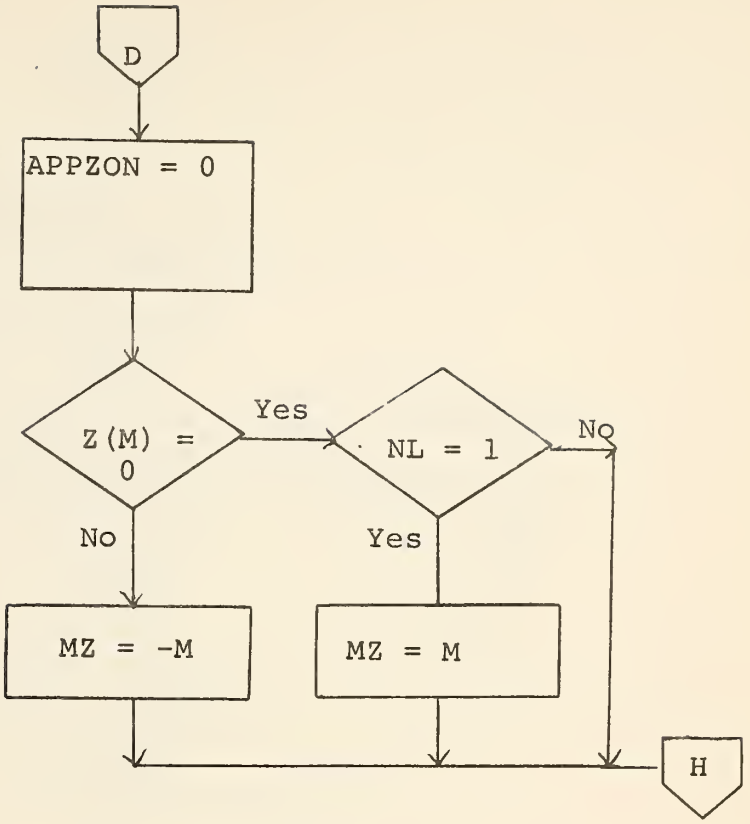


Figure 24. REZONE Program Flow Chart (Continued)

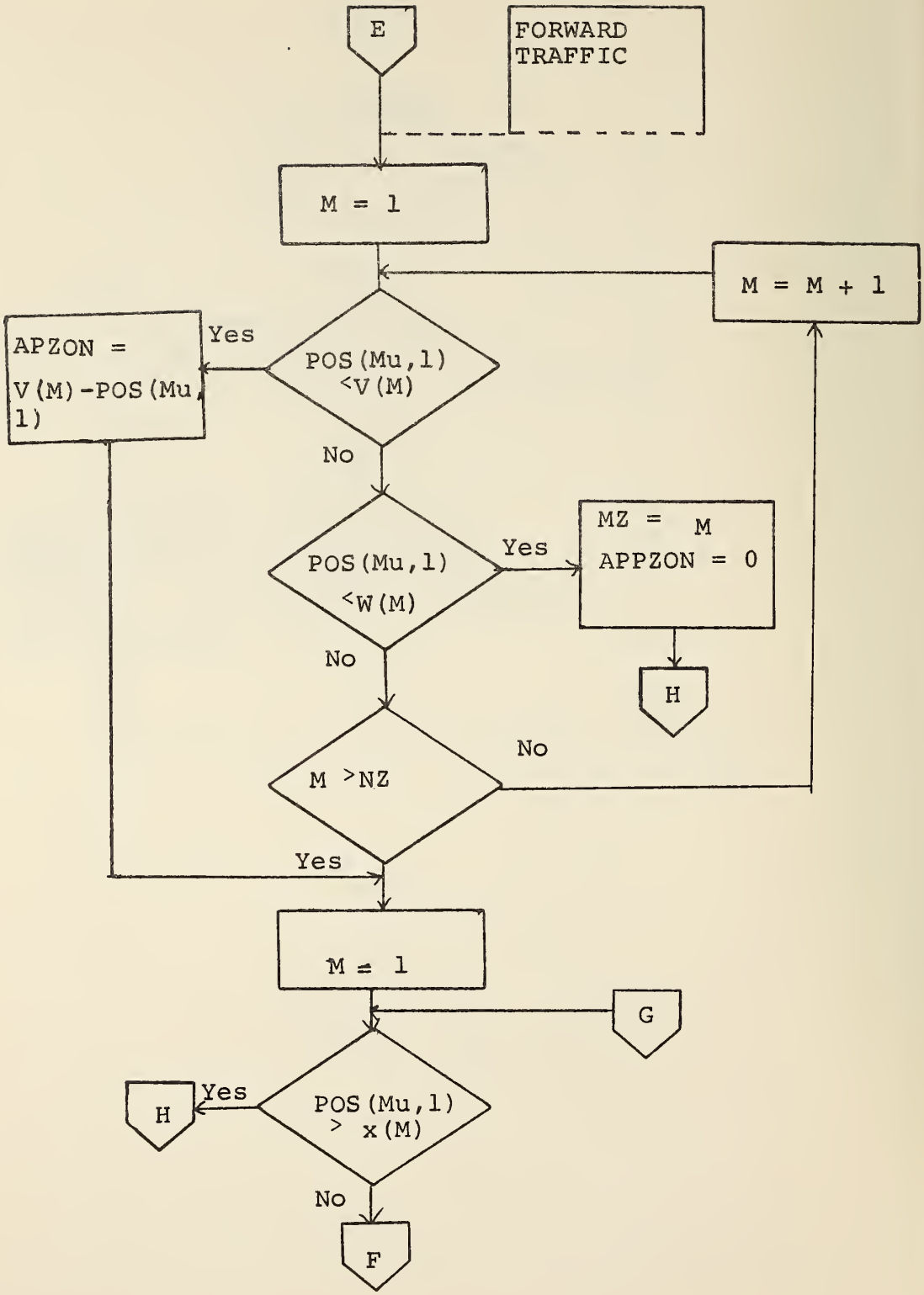


Figure 24. REZONE Program Flow Chart (Continued)

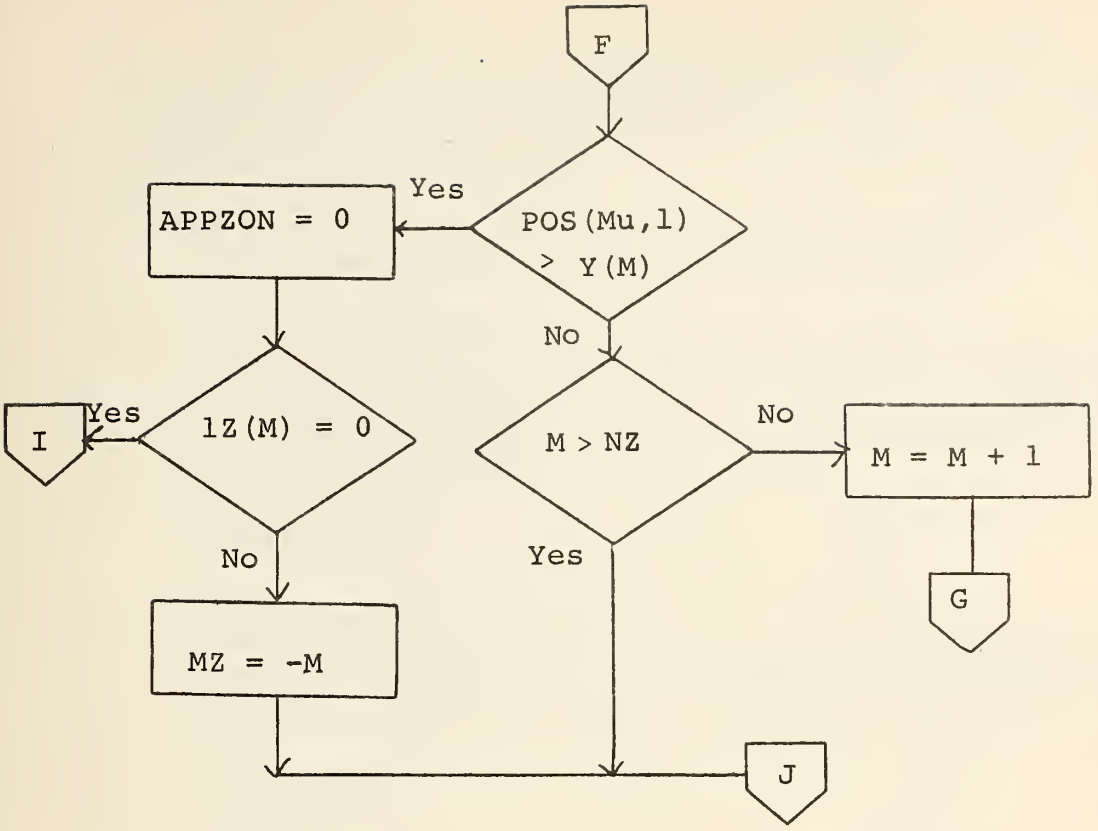


Figure 24. REZONE Program Flow Chart (Continued)

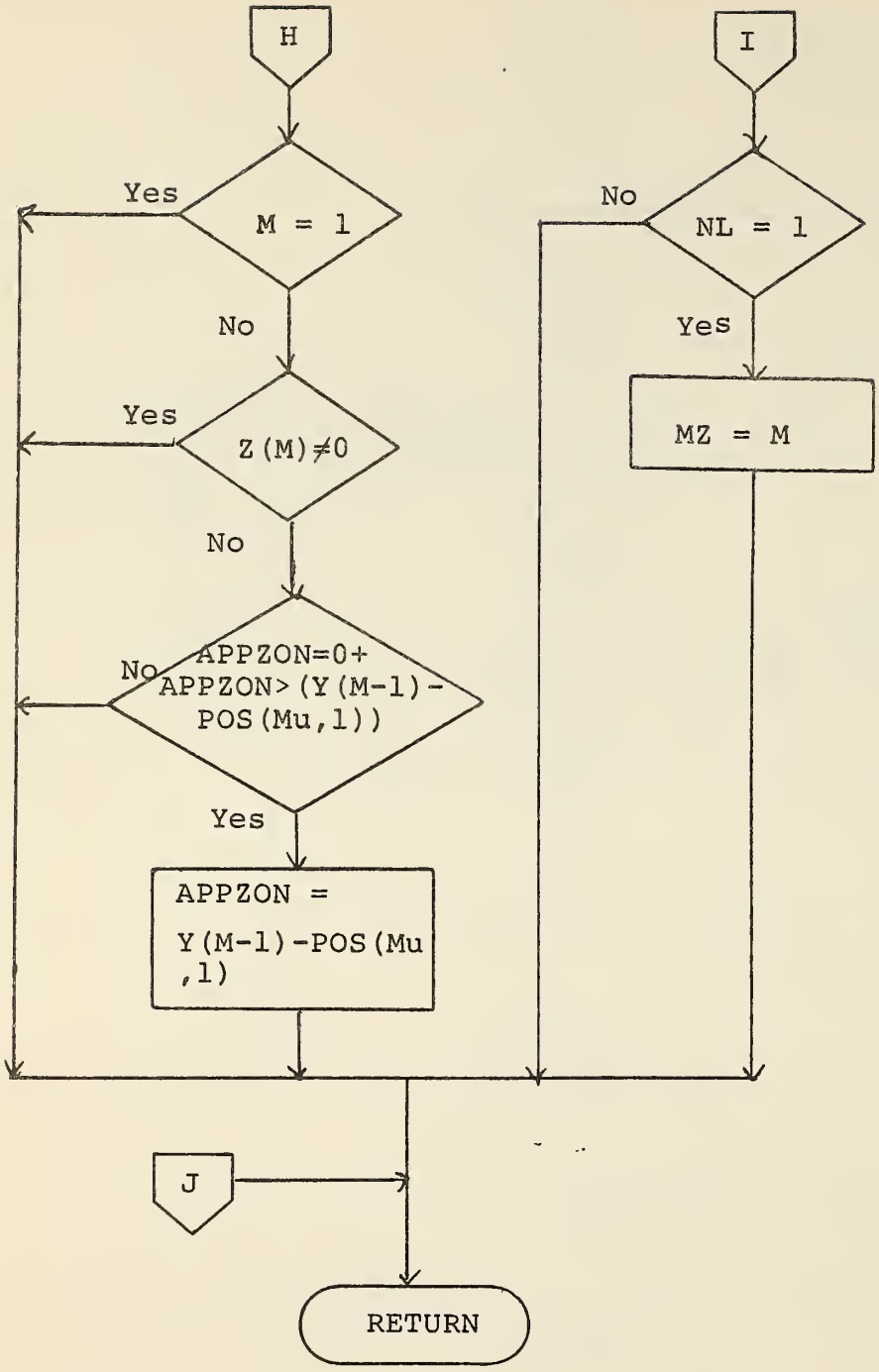


Figure 24. REZONE Program Flow Chart (Continued)

```
001             SUBROUTINE REZONE                                              00
                C
                C     DETERMINES WHETHER OR NOT VEHICLE IS IN RESTRICTED ZONE   00
                C RESTRICTED ZONES IN FORWARD DIRECTION START AT V AND END AT W,IN/ 00
                C REVERSE DIRECTION X AND Y ARE BEGINNING AND END OF ZONE. IZ=0 FOR 00
                C CURVE,=PER CENT GRADE OTHERWISE. IF VEHICLE IS IN UPGRADE MZ=M,  00
                C IF IN DOWNGRADE MZ=-M.                                       00
                C
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                  1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                  1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                  2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                  3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                  4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                  1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                  1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                  2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                  3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
006               LOGICAL DBUG
                C
007               APPZON=BRPOS                                                 00
008               MZ=0                                                         00
009               IF(ND.EQ.1.OR.LANE(MU).GT.0) GO TO 200                       00
                C FOR VEHICLE GOING IN REVERSE DIRECTION, COMPARE POSITION WITH X AND Y.00
                C IF VEHICLE IS IN RESTRICTED ZONE, SET MZ=TO ZONE INDEX AND APPZON=0. 00
010               DO 10 M=1,NZ                                                 00
011               IF(POS(MU)  .GT.X(M)) GO TO 30
012              20 IF(POS(MU)  .GT.Y(M)) GO TO 40
013              10 CONTINUE                                                   00
014               GO TO 50                                                     00
015              30 IF(X(M).EQ.0.) GO TO 50                                    00
016               APPZON=POS(MU)  -X(M)
017               GO TO 50                                                     00
018              40 MZ=M                                                       00
019               APPZON=0.                                                    00
020               GO TO 1000                                                   00
                C
                C IF VEHICLE IS NOT IN UPGRADE,IS IT IN DOWNGRADE OR CURVE.    00
                C
021              50 DO 80 M=1,NZ                                               00
022               IF(POS(MU)  .LT.V(M)) GO TO 60
023               IF(POS(MU)  .LT.W(M)) GO TO 70
024              80 CONTINUE                                                   00
```

```
025             GO TO 1000                                                      00
026            60 IF(M.EQ.1) GO TO 1000                                         00
027               IF (Z(M).NE.0.) GO TO 1000
028               IF (APPZON.EQ.0.0.AND.APPZON.GT.POS(MU)  -W(M-1)) APPZON=     00
                 $POS(MU)  -W(M-1)
029               GO TO 1000                                                    00
030            70 APPZON=0.                                                     00
031               IF (Z(M).EQ.0.) GO TO 90
032               MZ=-M                                                         00
033               GO TO 1000                                                    00
034            90 IF (NL.EQ.1)  MZ=M                                            00
035               GO TO 1000                                                    00
          C
          C SEARCH IN FORWARD DIRECTION                                         00
          C
036           200 DO 210 M=1,NZ                                                 00
037               IF(POS(MU)  .LT.V(M)) GO TO 220
038               IF(POS(MU)  .LT.W(M)) GO TO 230
039           210 CONTINUE                                                      00
040               GO TO 240
041           220 APPZON=V(M)-POS(MU)
042               GO TO 240                                                     00
043           230 MZ=M                                                          00
044               APPZON=0.                                                     00
045               GO TO 1000                                                    01
          C
          C SEARCH FOR DOWNGRADES AND CURVES.                                   01
          C
046           240 DO 280 M=1,NZ                                                 01
047               IF(POS(MU)  .GT.X(M)) GO TO 250
048               IF(POS(MU)  .GT.Y(M)) GO TO 260
049           280 CONTINUE                                                      01
050               GO TO 1000                                                    01
051           250 IF(M.EQ.1) GO TO 1000                                         01
052               IF (Z(M).NE.0.) GO TO 1000
053               IF (APPZON.EQ.0.0.AND.APPZON.GT.Y(M-1)-POS(MU)  ) APPZON=
                 $Y(M-1)-POS(MU)
054               GO TO 1000                                                    01
          C
055           260 APPZON=0.                                                     01
056               IF (Z(M).EQ.0.) GO TO 270
057               MZ=-M                                                         01
058               GO TO 1000                                                    01
059           270 IF(NL.EQ.1) MZ=M                                              01
060          1000 CONTINUE
061               IF(.NOT.DBUG)  GO TO 1002
062               WRITE (6,500) MZ,APPZON
063           500 FORMAT (1H ,I6,F10.2)                                         01
          C
064          1002 CONTINUE
065               RETURN                                                        01
066               END                                                           01
```

SORPOS

This subroutine reorders the indices of passed vehicles so that the forward vehicle is referenced first. Next cars are deleted off the end of the roadway, except that the last vehicle on the road is never removed. If the DBUG switch is set "TRUE", the buffer allocation tables and vehicle data tables are written out.

No subroutines are called by SORPOS.

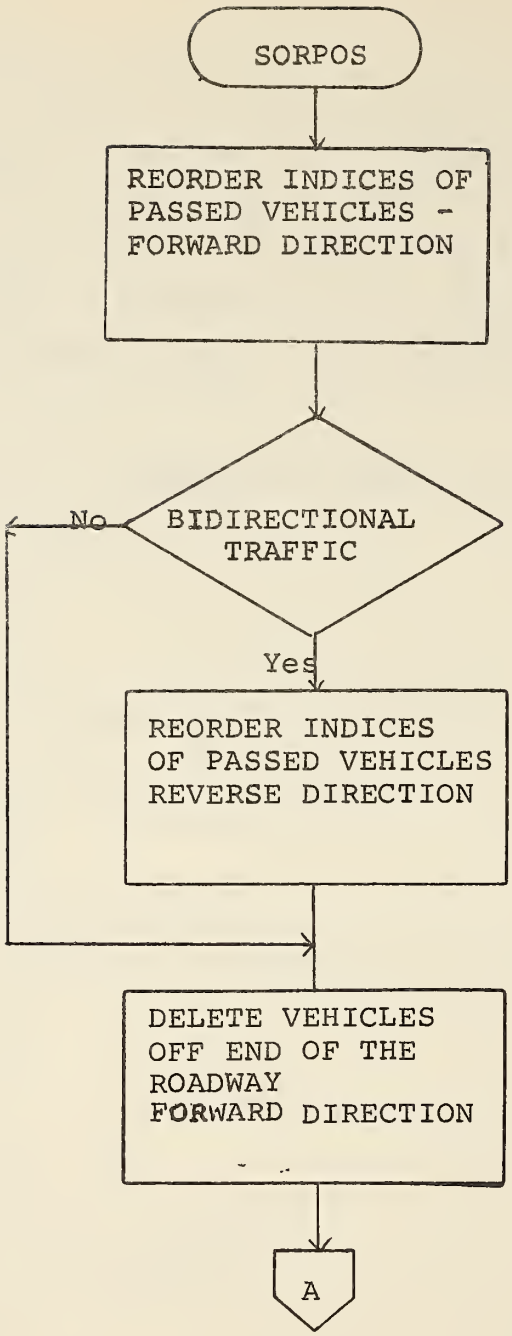


Figure 25. SORPOS Program Flow Chart

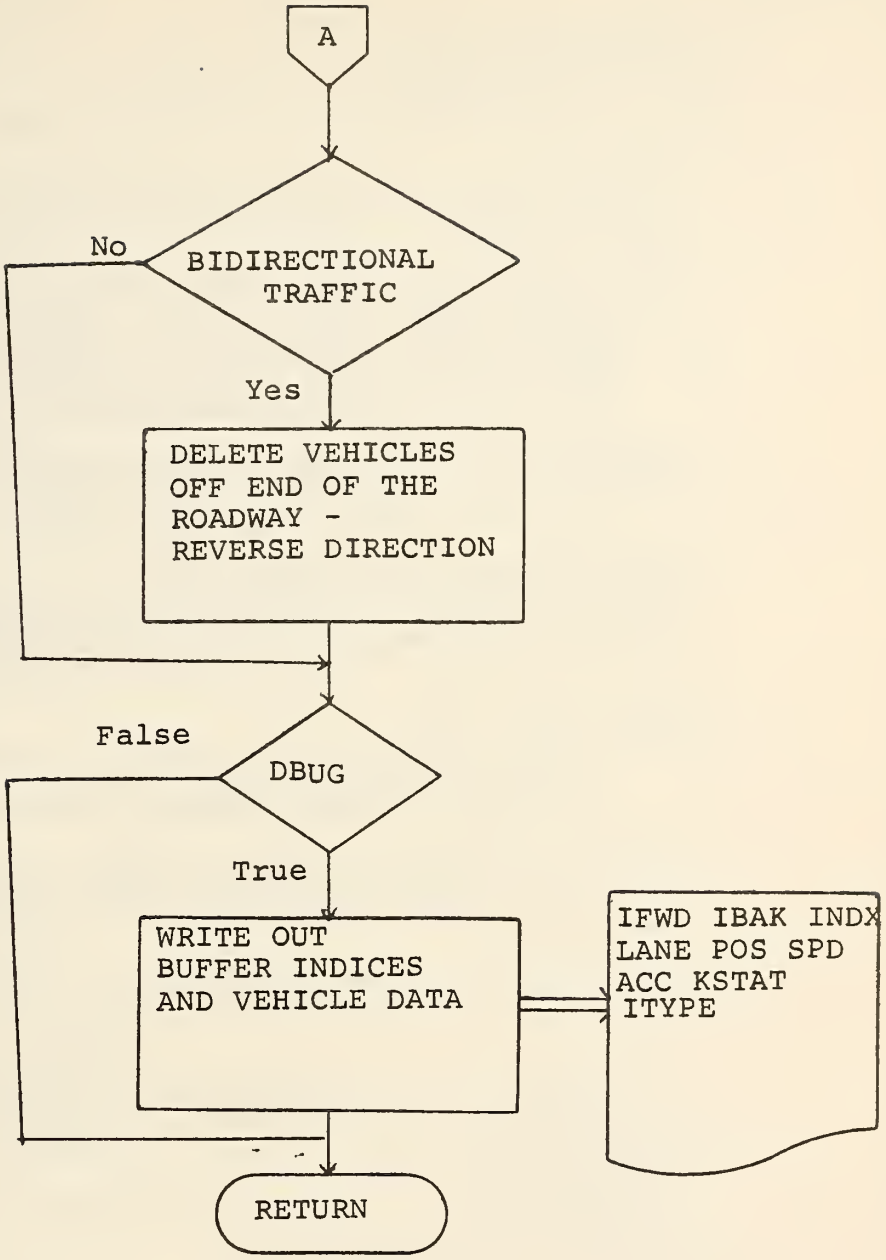


Figure 25. SORPOS Program Flow Chart (Continued)

```
001             SUBROUTINE SORPOS                                          01
          C ORDERS VEHICLES SO LEAD VEHICLE ALWAYS HAS SMALLER INDEX       01
          C THAN FOLLOWING VEHICLE.                                        01
          C   VEHICLES ARE DELETED OFF THE END OF THE ROADWAY
          C
          C   VEHICLE DATA
002             COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
               1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
          C
          C   BRIDGE, ROAD AND TIME DATA
003             COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
               1    OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
               2    TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
               3    XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
               4    NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
          C
          C   STATISTICAL DATA
004             COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
               1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
          C
          C   VEHICLE GENERATION DISTRIBUTION DATA
005             COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
               1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
               2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
               3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
          C
          C   BRIDGE LOADING DATA
006             COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
               1     XPOS(50), DXPOS(50), ACCLR(50),
               2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
          C
007             DIMENSION JFWD(200),JBAK(200),JNDX(200)
008             EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
               1           (JNDX(1),INDX(201))
009             LOGICAL DBUG
          C
          C  REORDER INDICIES OF PASSED VEHICLES
          C
010             IEND = 2
          C
011             IFD=IFWD(1)
012             IF (IFD.EQ.IBAK(1))  GO TO 35
013          25 I=INDX(IFD)
014             IF(IFD.GT.IEND)  IEND = IFD
015             IFD2=IFWD(IFD)
016             I2=INDX(IFD2)
017             IF(POS(I).LT.POS(I2)) GO TO 30
018          26 IFD=IFD2
019             IF(IFD.EQ.IBAK(1)) GO TO 35
020             GO TO 25
          C
          C  REORDER INDICIES
          C
021          30 INDX(IFD)=I2
```

```
022                INDX(IFD2)=I
023                GO TO 26
             C
             C
024             35 IF(ND.EQ.1) GO TO 50
             C
             C  REVERSE DIRECTION
             C
025                JEND = 2
026                JFD=JFWD(1)
027                IF (JFD.EQ.JBAK(1))  RETURN
028             40 J=JNDX(JFD)
029                IF (JFD.GT.JEND)  JEND = JFD
030                JFD2=JFWD(JFD)
031                J2=JNDX(JFD2)
032                IF(POS(J).GT.POS(J2)) GO TO 45
033             41 JFD=JFD2
034                IF(JFD.EQ.JBAK(1)) GO TO 50
035                GO TO 40
             C
             C  REORDER INDICIES
             C
036             45 CONTINUE
037                JNDX(JFD)=J2
038                JNDX(JFD2)=J
039                GO TO 41
             C
             C
040             50 CONTINUE
             C
             C  DELETE CARS OFF THE BRIDGE OR END OF ROADWAY
             C
041                KOUNT = 0
042                IFD=IFWD(1)
043                IF (IFD.EQ.IBAK(1)) GO TO 70
044             55 I=INDX(IFD)
045                ITY=ITYPE(I)
046                POSSI=POS(I)-VEHLEN(ITY)
047                IF(POSSI.LT.RDEND) GO TO 60
048                KOUNT = KOUNT + 1
049                IFD=IFWD(IFD)
050                IF (IFD.NE.IBAK(1) )  GO TO 55
051                IF(KOUNT.LE.1)  GO TO 70
052                IFD = IBAK(IFD)
             C
053             60 IF(IFD.EQ.IFWD(1)) GO TO 70
             C
             C
054                K=IBAK(IFD)
055                J=IFWD(1)
056                IFWD(K)=IFWD(IBAK(1))
057                IFWD(IBAK(1))=J
058                IFWD(IBAK(J))=IFD
             C
```

```
059               IBAK(IFWD(K))=IBAK(IFD)
060               IBAK(IFD)=IBAK(J)
061               IBAK(J)=IBAK(1)
                C
062              70 IF(ND.EQ.1) GO TO 100
063                 JFD=JFWD(1)
064                 IF (JFD.EQ.JBAK(1)) GO TO 100
065              75 J=INDX(JFD)
066                 ITY=ITYPE(J)
067                 POSSJ=POS(J)+VEHLEN(ITY)
068                 IF(POSSJ.GT.0.0) GO TO 80
069                 JFD=JFWD(JFD)
070                 GO TO 75
071              80 IF(JFD.EQ.JFWD(1)) GO TO 100
                C
                C
072                 K=JBAK(JFD)
073                 J=JFWD(1)
                C
074                 JFWD(K)=JFWD(JBAK(1))
075                 JFWD(JBAK(1))=J
076                 JFWD(JBAK(J))=JFD
                C
077                 JBAK(JFWD(K))=JBAK(JFD)
078                 JBAK(JFD)=JBAK(J)
079                 JBAK(J)=JBAK(1)
                C
080             100 CONTINUE
                C
081                 IF (.NOT.DBUG)  GO TO 1000
082                 WRITE(6,3032)
083                 DO 333 I=1,IEND
084              333 WRITE(6,3033) I, IFWD(I),IBAK(I), INDX(I), LANE(I), POS(I),
                    1     SPD(I,1), SPD(I,2), ACC(I), KSTAT(I), ITYPE(I)
085             3032 FORMAT('0     I  IFWD  IBAK  INDX  LANE  POSITION   SPEED 1   ',
                    1   'SPEED 2       ACC  STAT  TYPE')
086             3033 FORMAT (1X,5I6, 4F10.2, 2I6)
087                 IF(ND.EQ.1)  GO TO 1000
088                 WRITE(6,3034)
089                 DO 334 I=1,JEND
090              334 WRITE(6,3033) I, JFWD(I),JBAK(I), JNDX(I), LANE(I+200),
                    1  POS(I+200), SPD(I+200,1),SPD(I+200,2), ACC(I+200), KSTAT(I+200),
                    2     ITYPE(I+200)
091             3034 FORMAT('0     J  JFWD  JBAK  JNDX  LANE  POSITION   SPEED 1   ',
                    1   'SPEED 2       ACC  STAT  TYPE')
                C
092             1000 RETURN                                                    01
093                  END                                                       01
```

STAT

The STAT subroutine is called by CONTRO every simulated hour or by the main routine at the end of the simulation. STAT prints out the following data:

Platoons generated forward - by size
Platoons generated reverse - by size
Platoons captured on the bridge - by size
Vehicle type distribution
Vehicle load distribution - by specified increment

No subroutines are called by STAT.

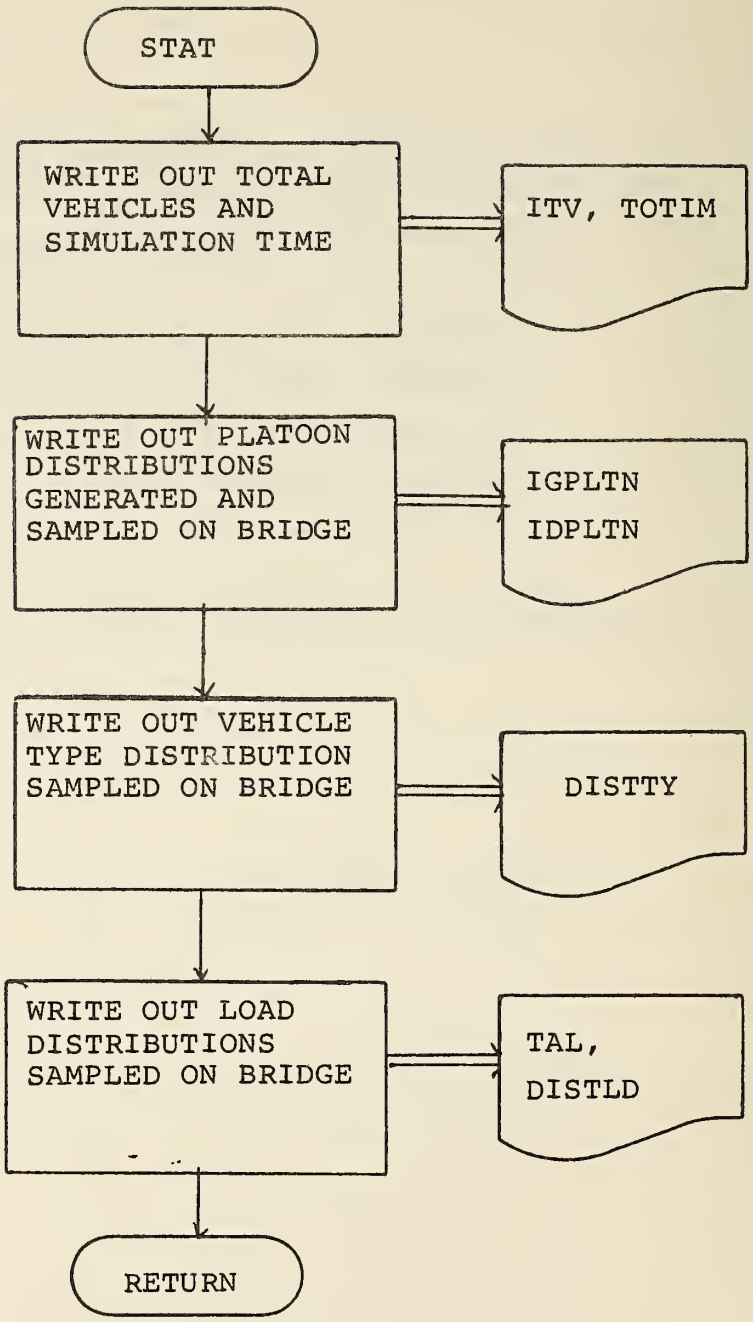


Figure 26. STAT Program Flow Chart

```
001             SUBROUTINE STAT
                C
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                 1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                 2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                 3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                 4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                 1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                 3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
006               INTEGER DISTTY, DISTLD
                C
007               WRITE (6,1009) ITV,TOTIM
008             1009 FORMAT('1TOTAL VEHICLES GENERATED =',I8,'    SIMULATED TIME =',
                    1    F12.0,' SECONDS')
009               WRITE (6,1010)
010             1010 FORMAT ('0PLATOON DISTRIBUTION',//,25X,'1       2       3',
                    1   '       4       5       6       7       8       9      10')
011               WRITE (6,1012) (IGPLTN(I,1),I=1,10)
012               IF (ND.EQ.1)  GO TO 11
013               WRITE (6,1013) (IGPLTN(I,2),I=1,10)
014              11 CONTINUE
015               WRITE(6,1011)  (IDPLTN(I),I=1,10)
                C
016             1011 FORMAT (' SAMPLED ON BRIDGE', 10I8)
017             1012 FORMAT (' GENERATED FORWARD', 10I8)
018             1013 FORMAT (' GENERATED REVERSE', 10I8)
                C
019               WRITE(6,1020)
020             1020 FORMAT('0TYPE DISTRIBUTION'//,6X,'1     2     3     4     5     ',
                    1  '6     7     8     9    10    11    12    13    14    15    ',
                    2  '16    17    18    19    20')
021               WRITE (6,1021)  (DISTTY(I),I=1,20)
022             1021 FORMAT (1X,20I6)
023               WRITE(6,1030)
024             1030 FORMAT ('0LOAD DISTRIBUTION',//)
025               DO 30 I=1,LT
026              30 WRITE(6,1032)  TAL(I),TAL(I+1), DISTLD(I)
027               WRITE(6,1033)  TAL(LV),  DISTLD(LV)
028             1032 FORMAT (1X, F6.0, '  TO ', F6.0, I8)
029             1033 FORMAT (6X, 'ABOVE ', F6.0, I8)
```

```
      C
030               RETURN
031               END
```

UPDATE

The UPDATE subroutine determines vehicle motion, that is, whether a vehicle must pass (call to PASPOS), whether a vehicle has completed a pass (call to PASTES) or whether a vehicle must follow (described below).

The desired spacing is calculated on the basis of the speed of the maneuvering vehicle and the length of the lead vehicle. This is a result of defining the vehicle position as that of the front bumper. At the beginning of the program a factor (f_2) is calculated such that:

$$f_2 = H_1/f_1 \tag{12}$$

where H_1 is the length of the first vehicle type entered and f_1 is 15 unless it is set otherwise. The factor f_1 has the dimensions of speed, and f_2 must be in seconds. The desired spacing is then:

$$G_d = v_M f_2 + H_L \tag{13}$$

When the difference of the positions of the maneuvering and lead vehicles is less than or equal to this value the former will attempt to pass. The minimum distance (D) permitted between the front bumper of the following car and the rear bumper of the lead car is 10 feet unless set otherwise.

If the vehicle is unable to pass, it is constrained to follow. Four situations are possible.

Both the present and original speed of the maneuvering vehicle is greater than the lead vehicle and the roadway is level.

The deceleration is:

$$a_M = v_M(v_L - v_M)/(x_L - x_M) \qquad (14)$$

If the vehicle is on an upgrade, the result of equation 15 is compared to the acceleration obtained by equation and the vehicle must decelerate at the smaller value.

The original speed of the maneuvering vehicle is less than the original speed of the lead vehicle, but the present speed of the former is greater than the latter. In this case the speed of the maneuvering vehicle is reduced to its orginal speed.

The original speed of the maneuvering vehicle is less than the present speed of the lead vehicle. No change is made and the vehicle is permitted to proceed as if it were in the free state.

If the distance between the two vehicles is 10 feet or less, $x_L - x_M = D$, the speed of the maneuvering vehicle is immediately reduced to that of the lead vehicle.

After the vehicle status, passing, free, or following is determined and lane changes made, the vehicle speeds and positions are updated as follows:

The new speed of a vehicle is calculated by the standard equation. The speed is not permitted to exceed the EXSPD value over the speed limit on the road.

$$v_{Mt} = v_{Mo} + \Delta T a_M \qquad (15)$$

where a_M is zero for vehicles in the free state.

The new position of a vehicle is determined by the old speed and acceleration:

$$x_{Mt} = x_{Mo} + \Delta T V_{Mt} + 1/2 (\Delta T)^2 a_M \qquad (16)$$

The value defines formally the location of the front bumper of the vehicle.

Bridge load data and run statistics are collected as vehicle positions are updated. Vehicles are reordered by position and the roadway and bridge load data are printed if the DBUG switch was set "TRUE."

The following subroutines are called by UPDATE:

REZONE
PASPOS
CALACC
PASTES
SORPOS
GRAPH

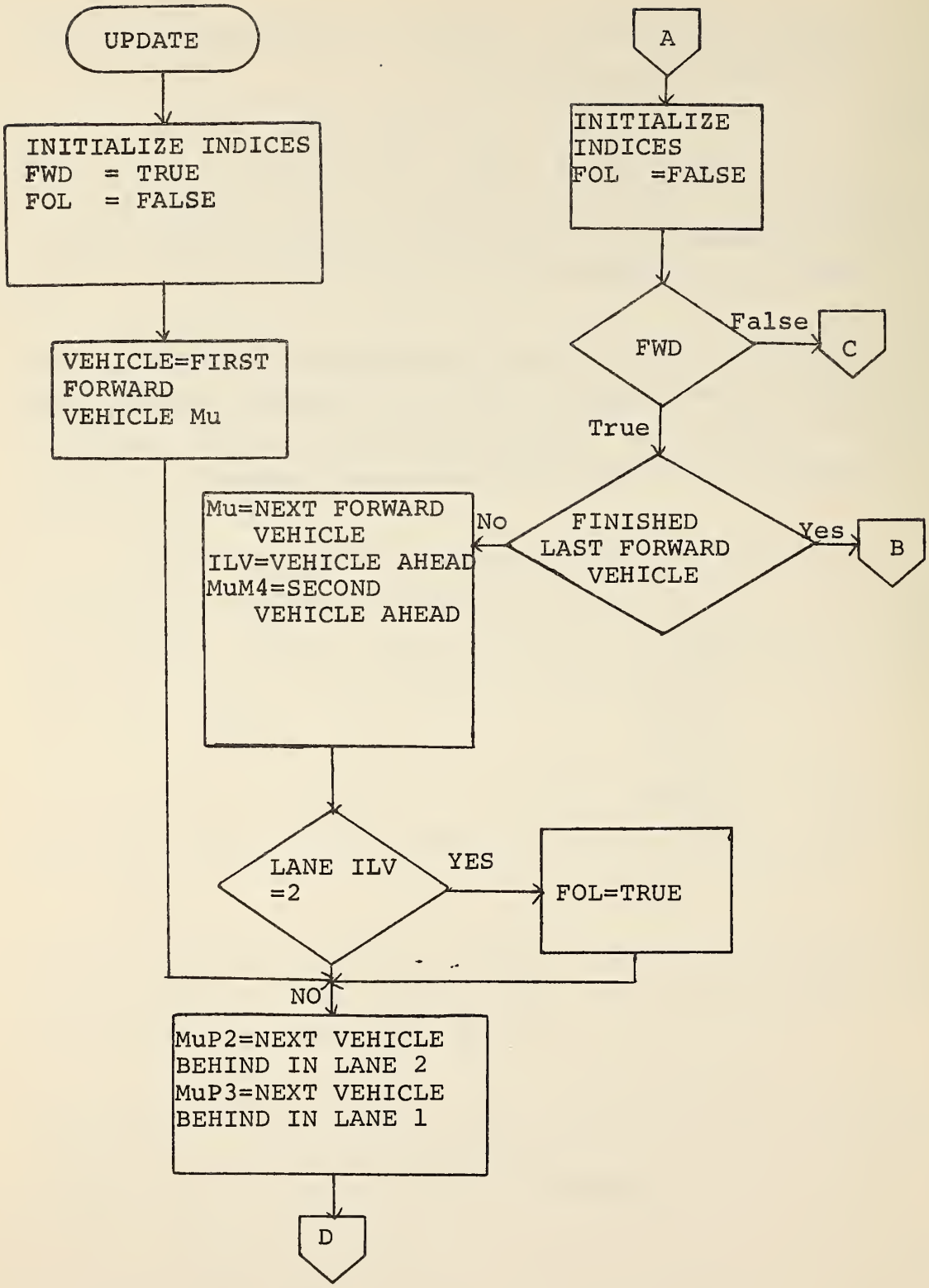


Figure 27. UPDATE Program Flow Chart

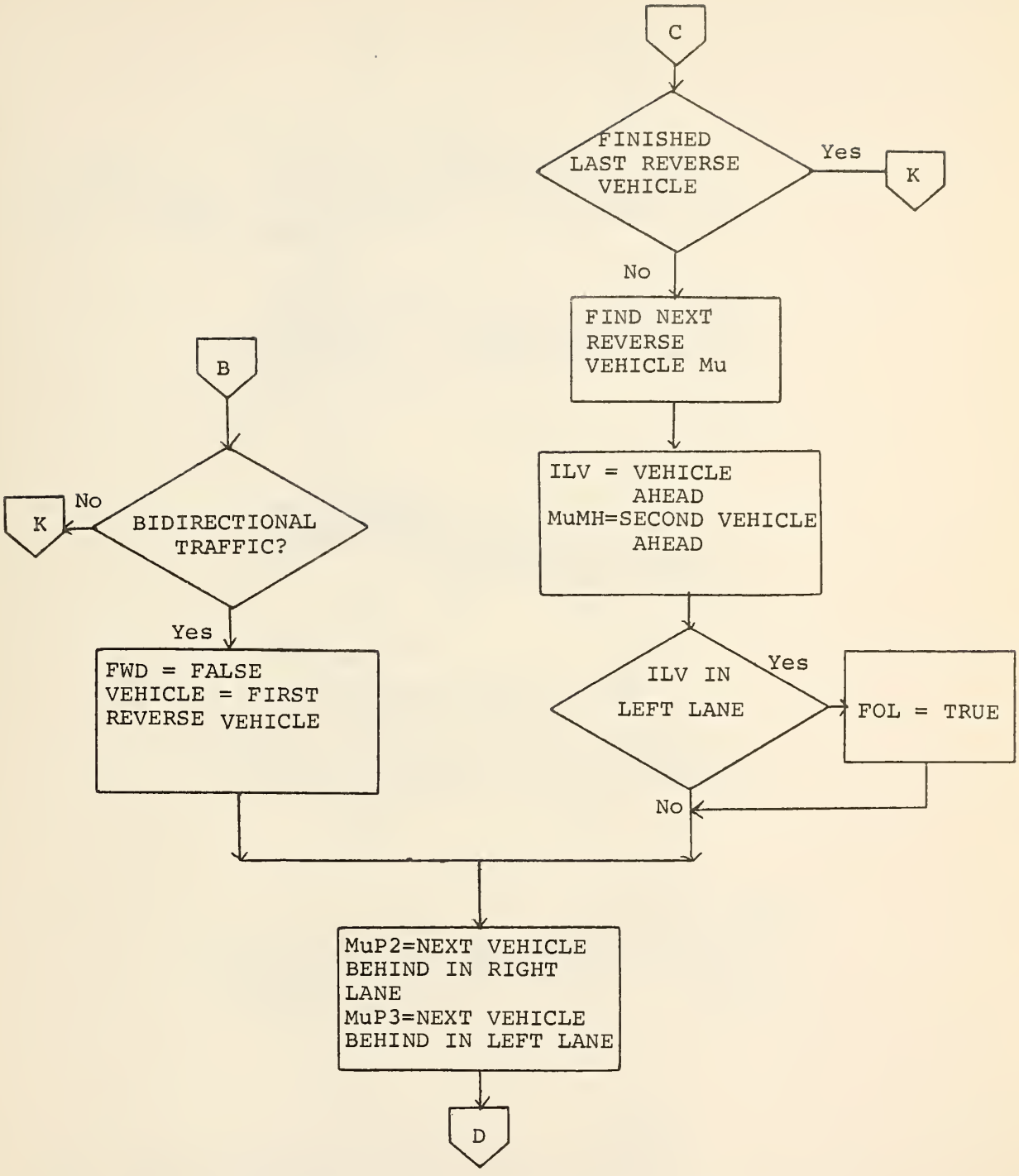


Figure 27. UPDATE Program Flow Chart (Continued)

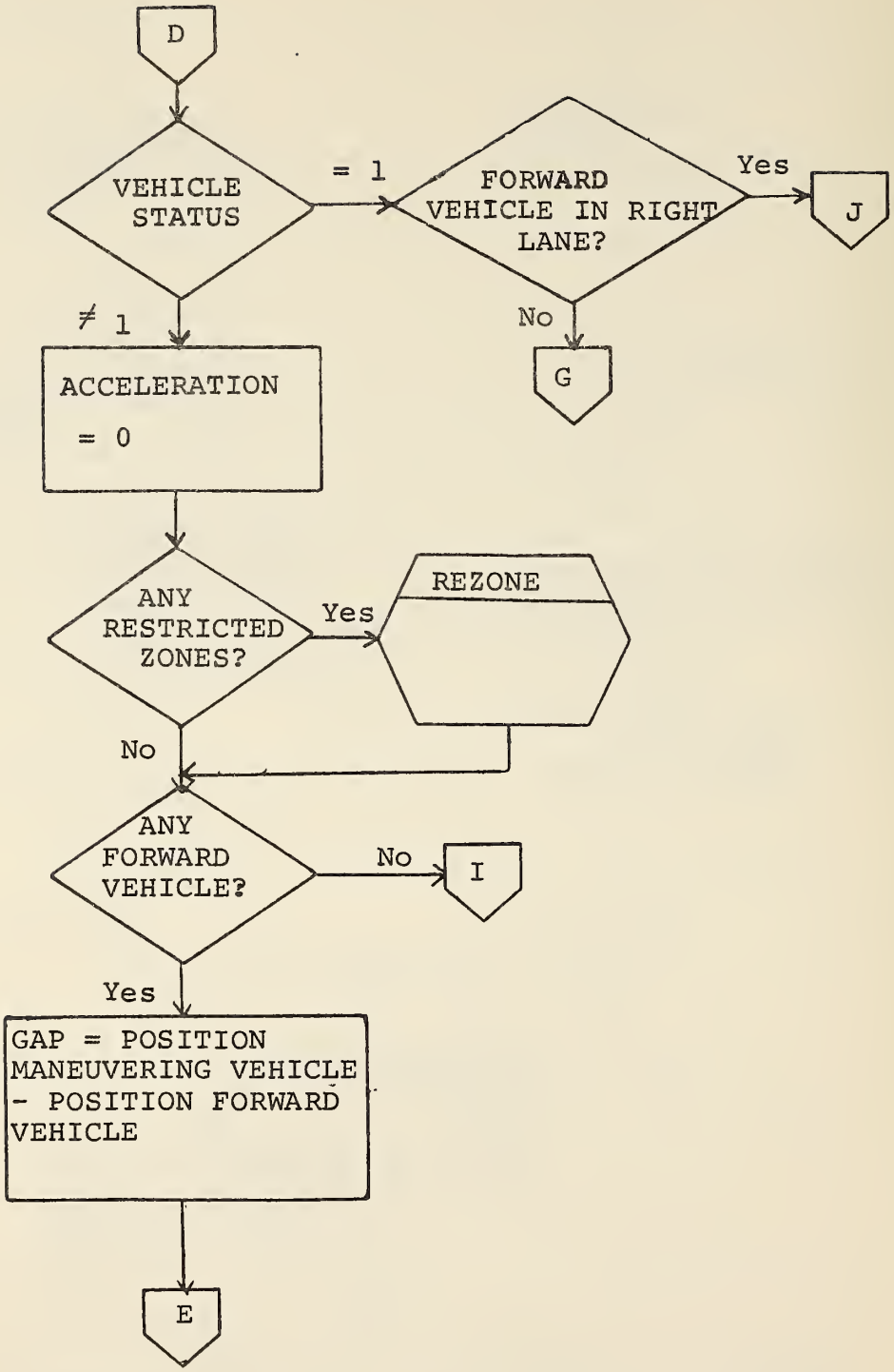


Figure 27. UPDATE Program Flow Chart (Continued)

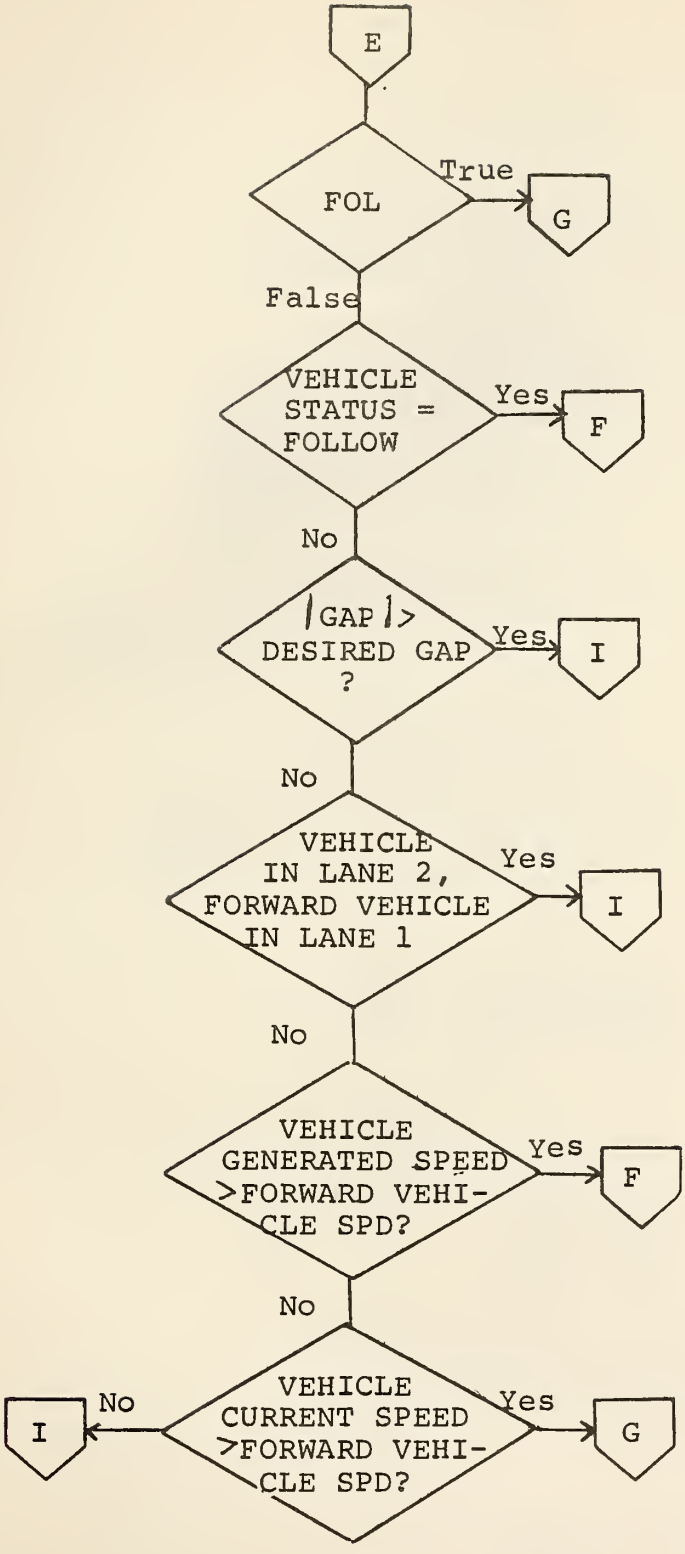


Figure 27. UPDATE Program Flow Chart (Continued)

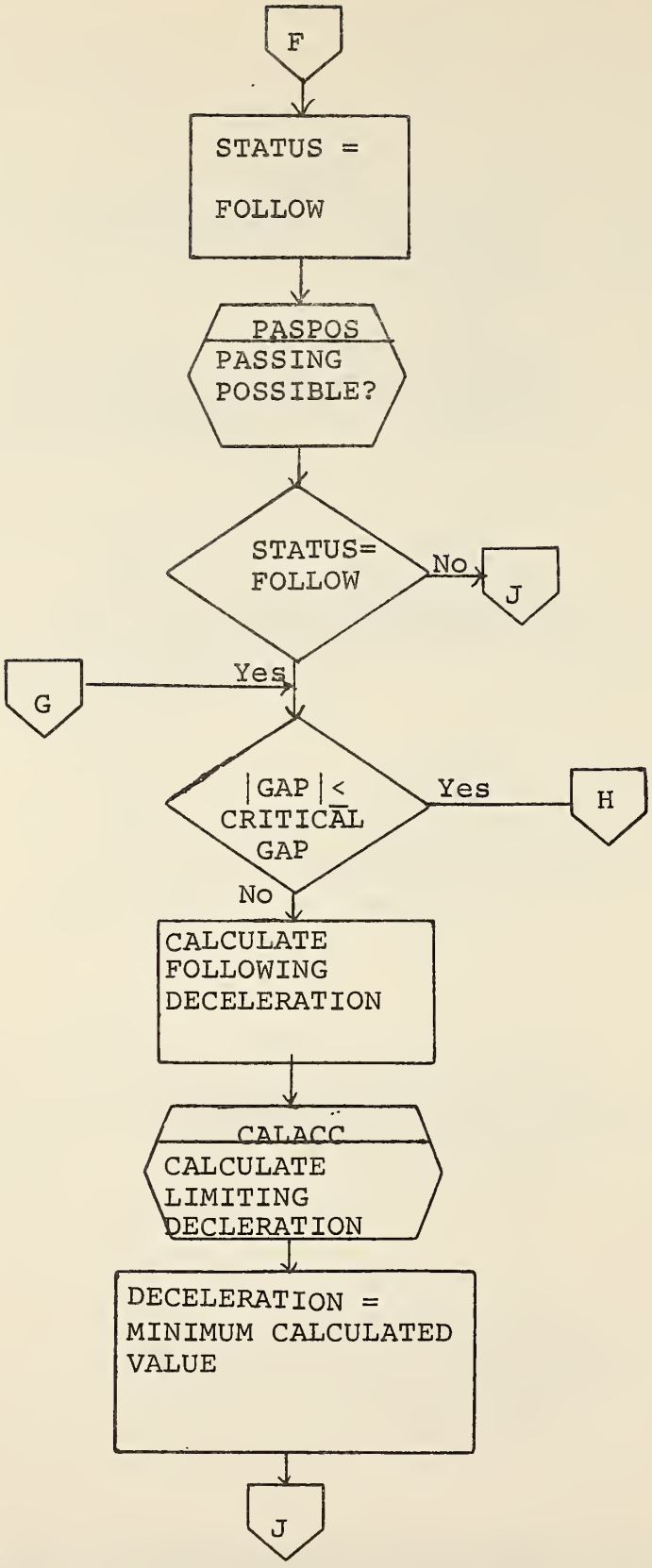


Figure 27. UPDATE Program Flow Chart (Continued)

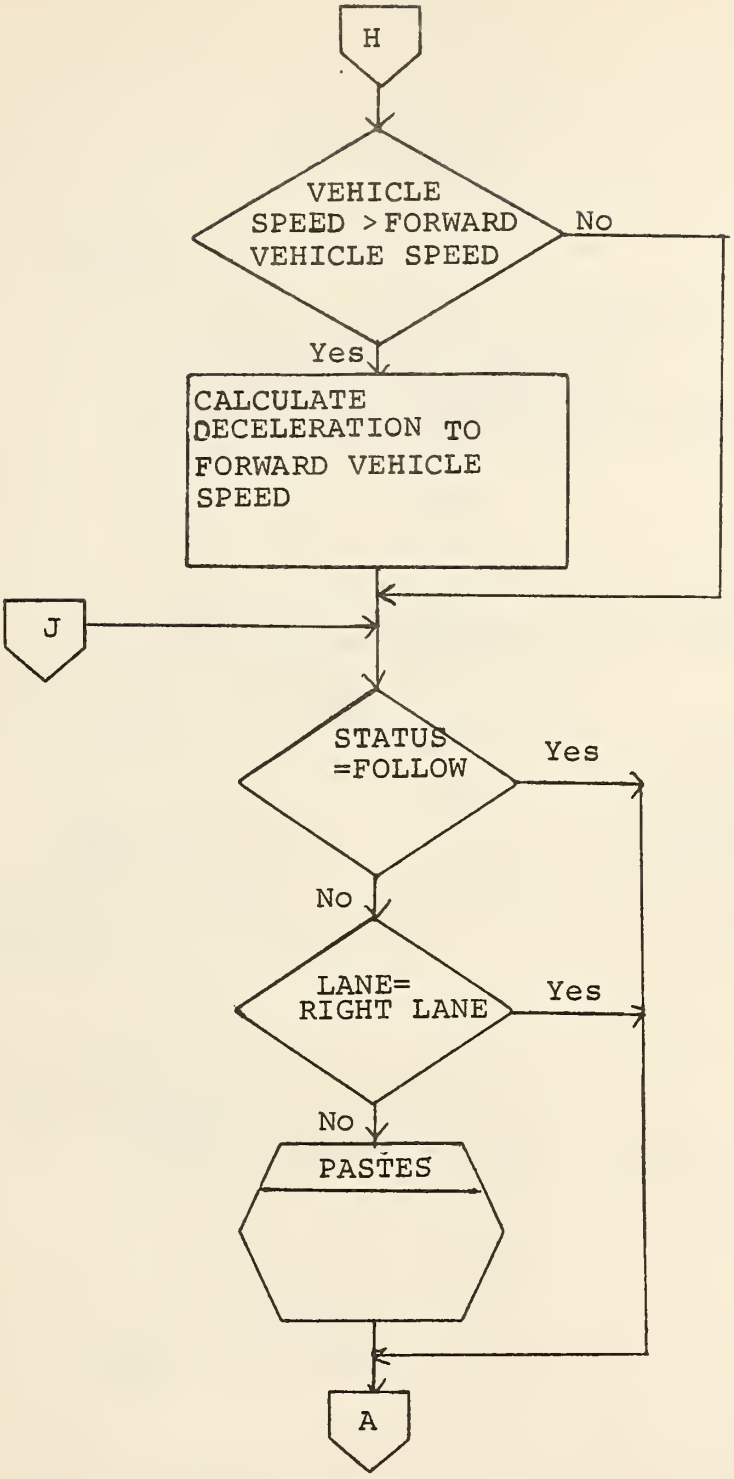


Figure 27. UPDATE Program Flow Chart (Continued)

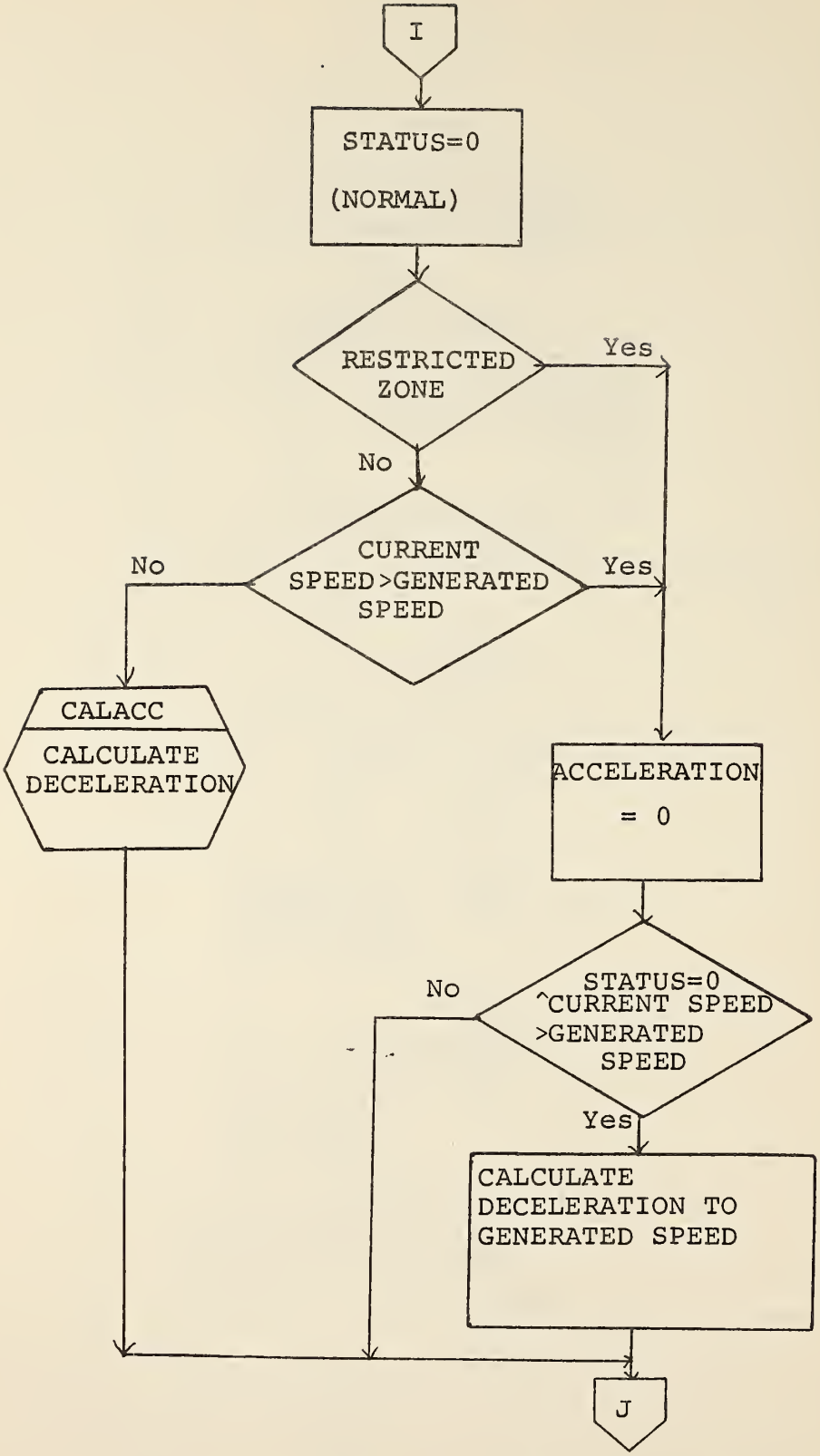


Figure 27. UPDATE Program Flow Chart (Continued)

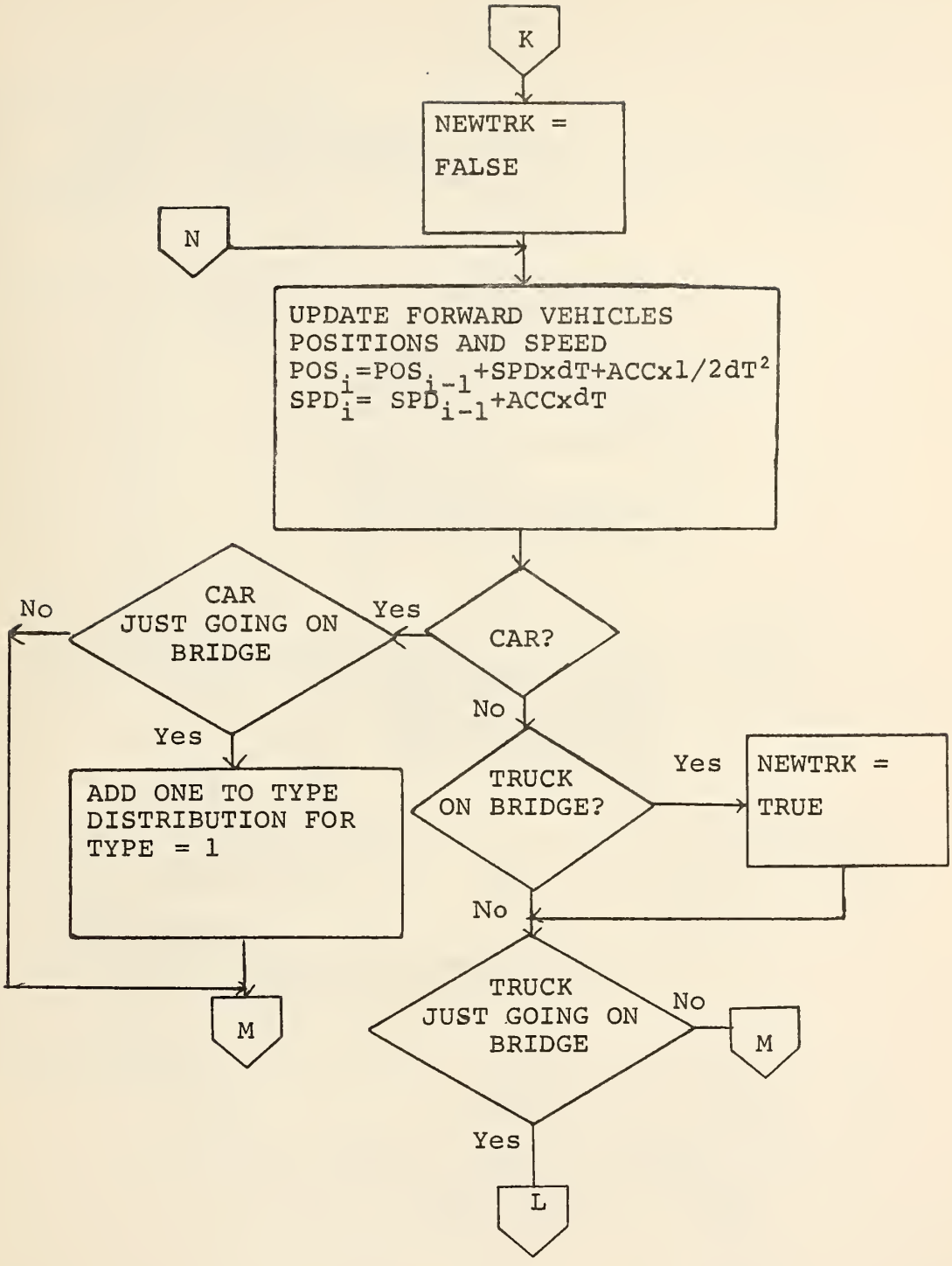


Figure 27. UPDATE Program Flow Chart (Continued)

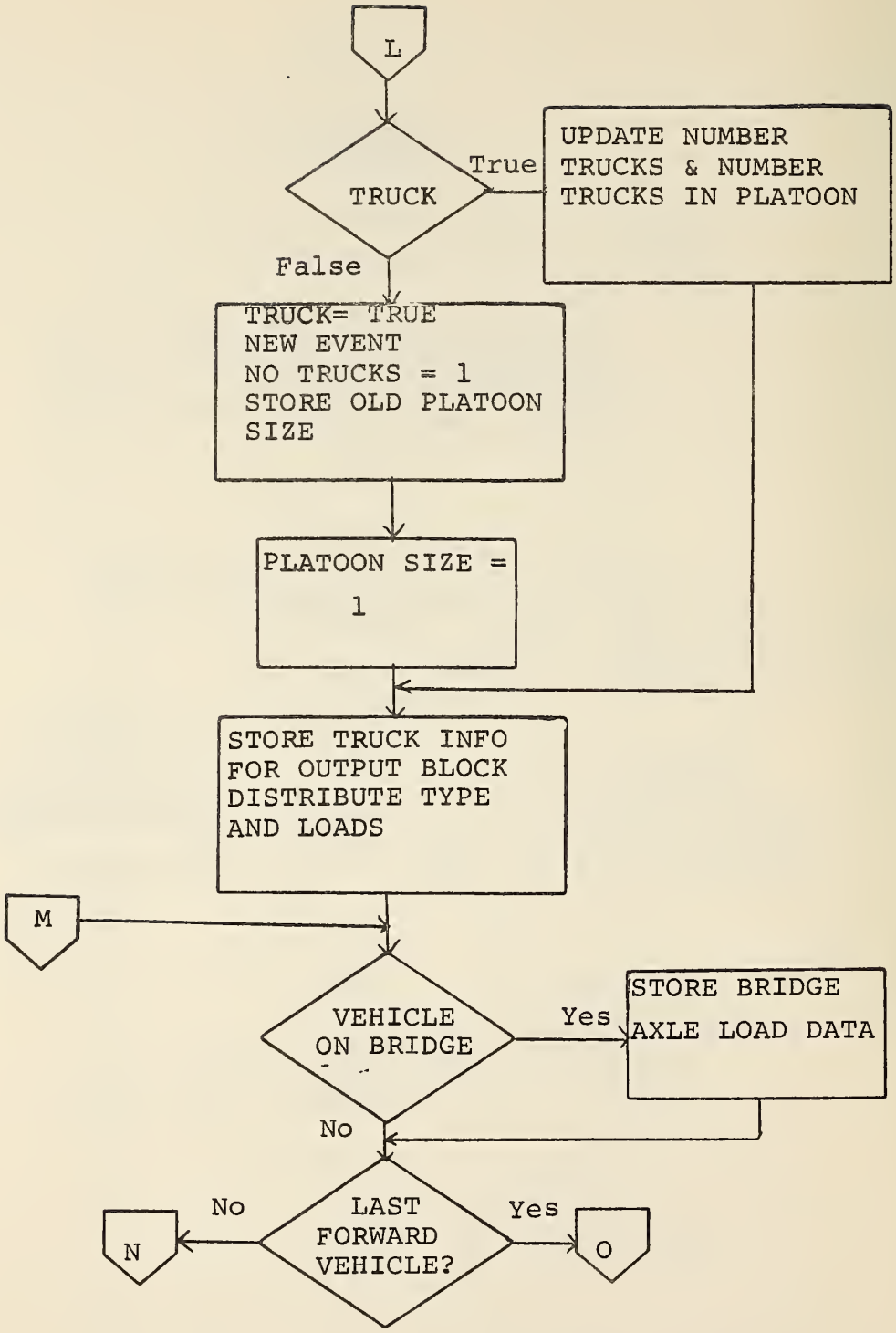


Figure 27. UPDATE Program Flow Chart (Continued)

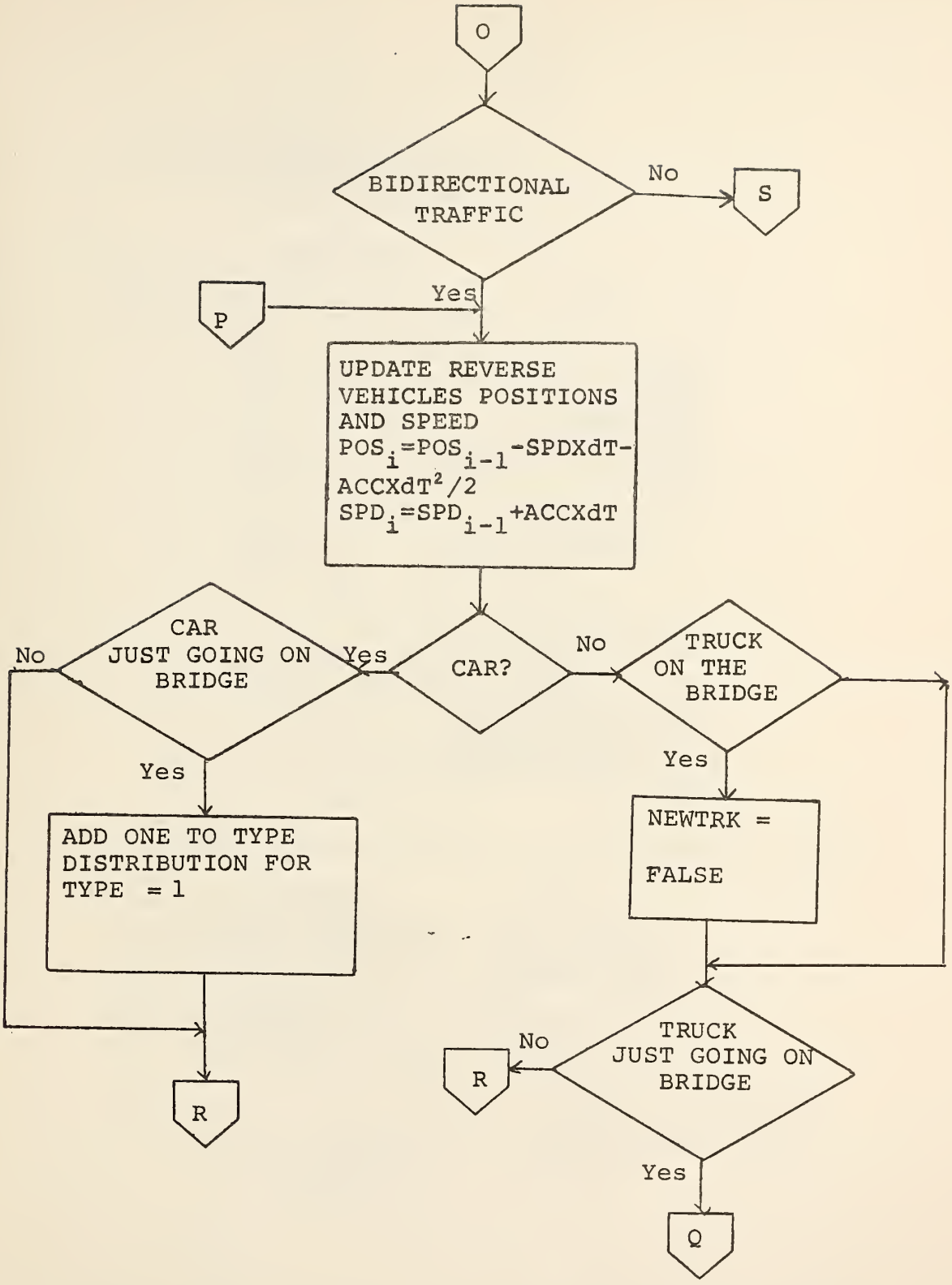


Figure 27. UPDATE Program Flow Chart (Continued)

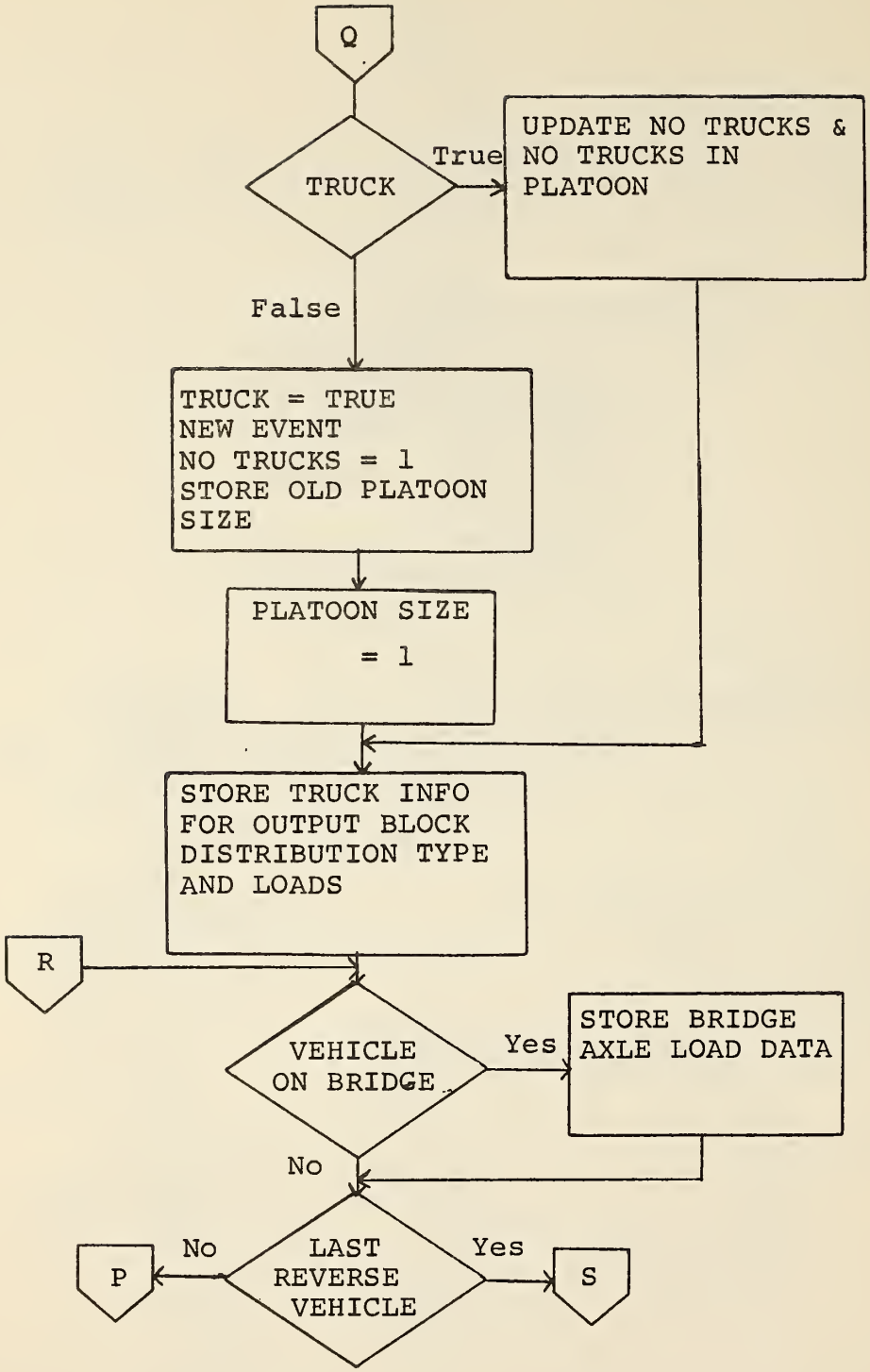


Figure 27. UPDATE Program Flow Chart (Continued)

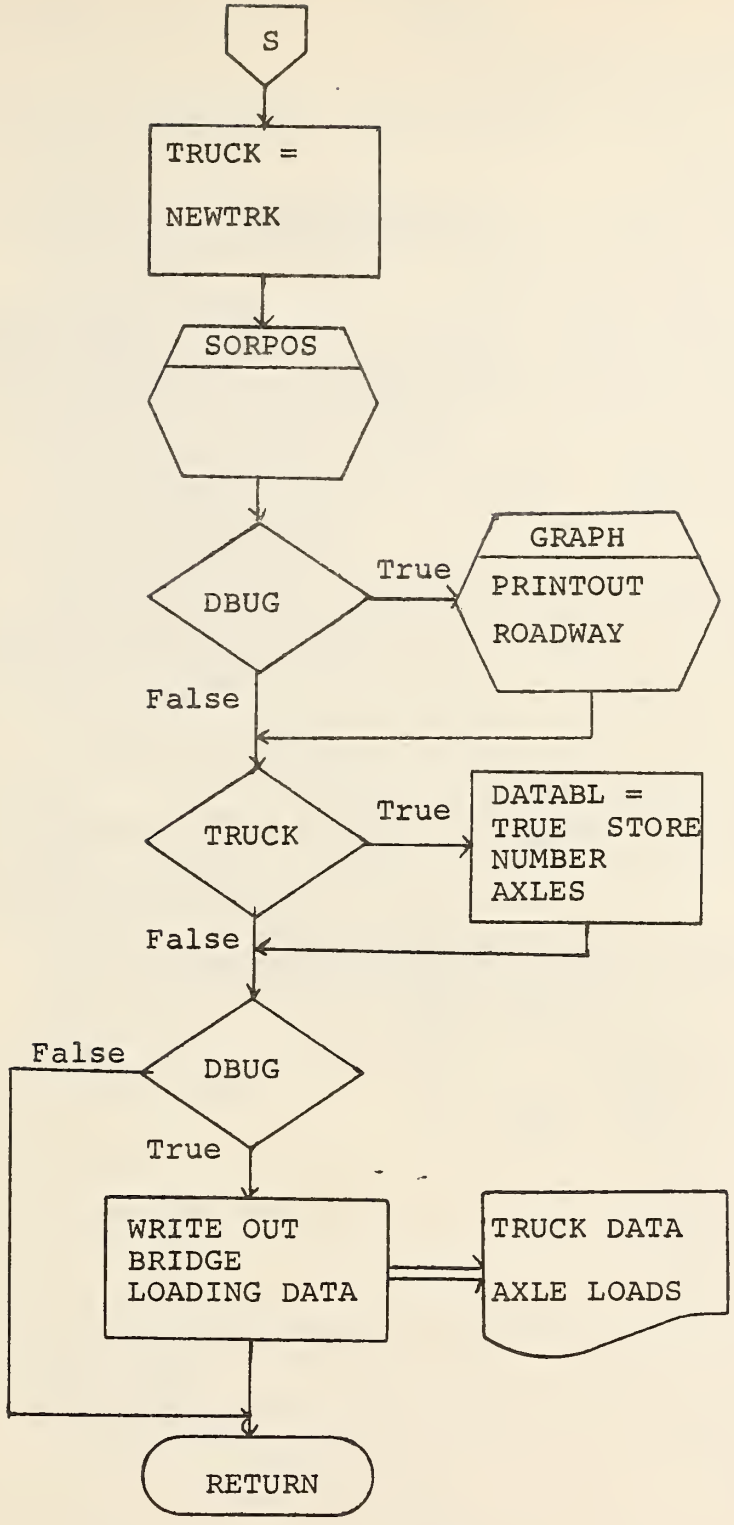


Figure 27. UPDATE Program Flow Chart (Continued)

```
001             SUBROUTINE UPDATE(DATABL)
                C
                C     MOVES VEHICLES FORWARD AT EACH TIME INCREMENT, CHECKS STATE.      01
                C
                C   VEHICLE DATA
002               COMMON ITYPE(400),WGT(400),SPD(400,2),POS(400),LANE(400),ACC(400)
                 1  ,KSTAT(400),     IFWD(400), IBAK(400), INDX(400)
                C
                C   BRIDGE, ROAD AND TIME DATA
003               COMMON BRLEN, BRST, BREND, APPZON, DESGAP, GLAD, GLEAD, CRIGAP,
                 1     OLDSPD, SPDIFF, HAFDEL, GAPFAC, HDFV, HDRV, TOTIM, BOUT,
                 2     TALINC, ACCEL, SPDLIM, SPDMAX, SPDMIN, TRKLIM, SPCK, FRTGAP,
                 3     XMIN, ILV, ITY, JOK, JOKE, LT, LV, MD, MU, MZ, ND, NGEN, NL,
                 4     NR, NTH, NZ, TIMLIM, BRPOS, DBUG, FIRST, RDEND, IOUT, NRAND
                C
                C   STATISTICAL DATA
004               COMMON  ITV,    PLATON(2), IPLTON(2), IGPLTN(10,2), IDPLTN(10)
                 1   ,DISTTY(20), DISTLD(50), TAL(51), MPLTON
                C
                C   VEHICLE GENERATION DISTRIBUTION DATA
005               COMMON/BLK/SDFAC,SAFDIS,NOAX(20),FT(4,6),AXWT(5,20),VEHLEN(20),
                 1AXPOS(5,20),SUBPER,AFR(20,2),HDTAB(40,2),SDTAB(20,20),AFS(20,2),
                 2WTAB(30,20),DPLTON(10,2),POWER(20),V(5),W(5),X(5),Y(5),Z(5)
                 3 ,FREQ(50,10),LHD(2),DELHD(2),LSP(20),DELSD(20),LWT(20),DELWT(20)
                C
                C   BRIDGE LOADING DATA
006               COMMON /BLK2/ SUMHR, DELTIM, IEVENT,NOAXL,NTRUK,LNUM(50),WEIT(50),
                 1     XPOS(50), DXPOS(50), ACCLR(50),
                 2     KTYPE(20), WGTT(20),SPDT(20), KLANE(20), TIMET(20)
007               COMMON /BLK3/ SUMHRX, DTL, IEVNTX, NOAXLX, NTRUKX, LAST,
                 1     LTYPE(20), WGTX(20),SPDX(20), LLANE(20), TIMEX(20),
                 2     XPOSX(50), LNUMX(50), WEITX(50), DXPOSX(50),ACCLRX(50)
                C
008               DIMENSION JFWD(200),JBAK(200),JNDX(200)
009               EQUIVALENCE (JFWD(1),IFWD(201)),(JBAK(1),IBAK(201)),
                 1           (JNDX(1),INDX(201))
010               LOGICAL TRUCK, PLATON, DBUG
011               LOGICAL FWD, FOL
012               LOGICAL NEWTRK
013               LOGICAL DATABL, FIRST, LAST
014               INTEGER DISTTY, DISTLD
                C
015               IFD = IFWD(1)
016               MU = INDX(IFD)
017               FWD = .TRUE.
018               ILV = 0
019               FOL = .FALSE.
020               MUP2 = 0
021               MUP3 = 0
022               MUM4 = 0
023               DATABL = .FALSE.
024               GO TO 820
                C
025             900 CONTINUE
```

```
      C
      C
026           OLDSPD=SPD(MU,1)                                          01
027           ITY = ITYPE(MU)                                           01
028           IF (KSTAT(MU).EQ.1)  GO TO 760
029           ACC(MU) = 0.0
030           MZ=0                                                      01
031           IF (NZ.NE.0) CALL REZONE                                  01
032           IF (ILV.EQ.0)  GO TO 50
033           GAP = POS(MU) - POS(ILV)
034           SPCK=ABS(GAP)                                             01
035       660 ITYL=ITYPE(ILV)                                           01
036           IF (FOL)  GO TO 35
037           IF (KSTAT(MU).LT.0) GO TO 20                              01
038        30 DESGAP=ABS(GAPFAC*SPD(MU,1)) +VEHLEN(ITYL)                01
039           IF(SPCK.GT.DESGAP) GO TO 50                               01
040           IF (LANE(MU).EQ.2.AND.LANE(ILV).NE.2)  GO TO 50
041           A=ABS (SPD(ILV,1))                                        01
042           B=ABS(SPD(MU,2))                                          01
043       800 IF (B.GT.A) GO TO 20                                      01
044           A=ABS(SPD(ILV,1))                                         01
045           B=ABS(SPD(MU,1))                                          01
046           IF(B-A) 50,50,35                                          01
047        20 KSTAT(MU)=-1                                              01
048           CALL PASPOS(MUM4, MUP2, MUP3)
      C       IF IN FOLLOWING STATE, IS PASSING POSSIBLE.              01
049           IF (KSTAT(MU).GT.-1) GO TO 770
      C       IF PASSING NOT POSSIBLE, FOLLOW.                         01
      C
      C    FOLLOW
      C      DETERMINES NEW SPEED OF VEHICLE WHICH IS CONSTRAINT TO FOLLOW.  00
      C
050        35 CONTINUE
051           CRIGAP=VEHLEN(ITYL) + SAFDIS                              00
052           IF(SPCK.LE.CRIGAP) GO TO 12
053           IF(ABS(SPD(MU,1)).LE.ABS(SPD(ILV,1)))  GO TO 750
054           DECEL=SPD(MU,1)*(SPD(MU,1)-SPD(ILV,1))/(POS(MU) - POS(ILV))
055           IF(MZ.LE.0) GO TO 32
056           CALL CALACC
057           IF (ABS(ACC(MU)).LE.ABS(DECEL)) GO TO 770
058        32 ACC(MU) = DECEL
059           GO TO 770
      C
060        12 CONTINUE
061           IF (SPD(MU,1).GT.SPD(ILV,1)) ACC(MU) = (SPD(ILV,1)-SPD(MU,1))/
             1   DELTIM
062           GO TO 753
      C
063        50 KSTAT(MU)=0                                               01
064           IF(MZ.LE.0.OR.SPD(MU,1).GE.SPD(MU,2)) GO TO 750           01
065       700 CALL CALACC                                               01
066           GO TO 753
      C       IF PASSING POSSIBLE, ACCELERATE                          01
      C
```

```
067          750 ACC(MU) = 0.0
068          751 IF (KSTAT(MU).EQ.0.AND.ABS(OLDSPD).GT.ABS(SPD(MU,2)))              01
                1  GO TO  752
069              GO TO 770
             C
070          760 CONTINUE
071              IF(ILV.EQ.0)  GO TO 770
072              IF(LANE(ILV).EQ.1.OR.LANE(ILV).EQ.-NL)  GO TO 770
073              GAP = POS(MU) - POS(ILV)
074              SPCK=ABS(GAP)
075              ITYL=ITYPE(ILV)
076              GO TO 35
             C
077          752   ACC(MU) = (SPD(MU,2) - SPD(MU,1)) /DELTIM
078          753 CONTINUE
079          770 CONTINUE
080          780 IF (KSTAT(MU).LT.0) GO TO 10                                       01
081              IF (LANE(MU).EQ.1.OR.LANE(MU).EQ.-NL) GO TO 10                     01
082          790 CONTINUE
083              CALL PASTES (MUP2)
             C       IS PASS COMPLETED.                                            01
             C
084           10 CONTINUE                                                         01
085              ILV = 0
086              FOL = .FALSE.
087              MUP2 = 0
088              MUP3 = 0
089              MUM4 = 0
             C
090              IF (.NOT.FWD)  GO TO 860
091              IF (IFD.EQ.IBAK(1))  GO TO 850
092              JLV = IFD
093              IFD = IFWD(IFD)
094              ILV = MU
095              MU = INDX(IFD)
096              IF(LANE(ILV).EQ.2)  FOL = .TRUE.
             C
             C    FIND 2ND VEHICLE AHEAD
097              JLV =IBAK(JLV)
098              IF (JLV.NE.1) MUM4 = INDX(JLV)
             C
             C    FIND VEHICLES BEHIND
             C
099          820 IBK = IFWD(IFD)
             C
100              IF(IFD.EQ.IBAK(1))  GO TO 900
101          802 IBX = INDX(IBK)
102              IF(LANE(IBX).EQ.2)  GO TO 804
103              IF (MUP2.EQ.0)  MUP2 = IBX
             C
104          803 CONTINUE
105              IF (IBK.EQ.IBAK(1))  GO TO 900
106              IBK = IFWD(IBK)
107              GO TO 802
```

```
      C
108     804 IF (MUP3.EQ.0)  MUP3 = IBX
109         IF (MUP2.EQ.0)  GO TO 803
110         GO TO 900
      C
111     850 IF (ND.EQ.1)  GO TO 890
      C
112         FWD = .FALSE.
113         IFD = JFWD(1)
114         MU = JNDX(IFD)
115         GO TO 880
116     860 CONTINUE
117         IF (IFD.EQ.JBAK(1))  GO TO 890
118         JLV = IFD
119         IFD = JFWD(IFD)
120         ILV = MU
121         MU = JNDX(IFD)
122         IF (LANE(ILV).EQ.-1)  FOL = .TRUE.
      C
      C   FIND 2ND VEHICLE AHEAD
123         JLV = JBAK(JLV)
124         IF (JLV.NE.1)  MUM4 = JNDX(JLV)
      C
      C   FIND VEHICLE BEHIND
125     880 IBK = JFWD(IFD)
126         IF(IFD.EQ.JBAK(1)) GO TO 900
127     882 IBX = JNDX(IBK)
128         IF (LANE(IBX).EQ.-1)  GO TO 884
129         IF(MUP2.EQ.0)  MUP2 = IBX
130     883 CONTINUE
131         IF (IBK.EQ.JBAK(1))  GO TO 900
132         IBK = JFWD(IBK)
133         GO TO 882
      C
134     884 IF (MUP3.EQ.0)  MUP3 = IBX
135         IF (MUP2.EQ.0)  GO TO 883
136         GO TO 900
      C
137     890 CONTINUE
      C
138         NEWTRK=.FALSE.
      C
      C  UPDATE POSITIONS
      C
139         K = 1
140         IFD=IFWD(1)
141       5 I=INDX(IFD)
142         MU = I
143         POSOLD=POS(I)
144         IF (ABS(ACC(MU)).GT.ACCEL)  ACC(MU) = SIGN(ACCEL,ACC(MU))
145         POS(I)=POS(I)+SPD(I,1)*DELTIM+HAFDEL*ACC(I)*DELTIM
146         SPD(MU,1)=ACC(MU)*DELTIM + SPD(MU,1)
147         IF(ABS(SPD(MU,1)).LT.SPDMIN)  SPD(MU,1) = SIGN(SPDMIN,SPD(MU,1))
148         IF(ABS(SPD(MU,1)).GT.SPDMAX) SPD(MU,1)=SIGN(SPDMAX,SPD(MU,1))      01
```

```
149               IF(ITYPE(I).EQ.1) GO TO 99
150               IF(POS(I).GT.BRST.AND.POSOLD.LT.BRENDJ  NEWTRK = .TRUE.
151               IF(POSOLD.GT.BRST   .OR.POS(I).LT.BRST)    GO TO 9
152               IF(TRUCK) GO TO 7
                C
153               TRUCK=.TRUE.
154               IEVENT = IEVENT + 1
155               NTRUK = 1
156               IF (MPLTON.GT.10)  MPLTON = 10
157               IDPLTN(MPLTON) = IDPLTN(MPLTON) + 1
158               MPLTON = 1
159               GO TO 8
160            99 CONTINUE
161               IF(POSOLD.GT.BRST   .OR.POS(I).LT.BRST)    GO TO 9
162               DISTTY(1) = DISTTY(1) + 1
163               GO TO 9
164             7 CONTINUE
165               MPLTON = MPLTON + 1
166               NTRUK = NTRUK + 1
                C
                C  DISTRIBUTE TYPE AND LOADS
                C
167             8 CONTINUE
168               ITY=ITYPE(I)
169               KTYPE(NTRUK) = ITY
170               WGTT(NTRUK) = WGT(I)
171               SPDT(NTRUK) = SPD(I,1)
172               KLANE(NTRUK) = LANE(I)
173               TIMET(NTRUK) = TOTIM
                C
174               DISTTY(ITY)=DISTTY(ITY)+1
175               ILD=WGT(I)/TALINC + 1
176               IF (ILD.GT.LV) ILD = LV
177               DISTLD(ILD)=DISTLD(ILD)+1
178             9 CONTINUE
                C
                C  COLLECT BRIDGE LOADING DATA
                C
179               IF(POS(I).LT.BRST)    GO TO 120
180               ITY=ITYPE(I)
181               NAX=NOAX(ITY)
182               POSS = POSOLD - VEHLEN(ITY)
183               IF(POSS.GT.BREND)  GO TO 120
184               DO 110 IAX=1,NAX
185               LNUM(K)=LANE(I)
186               WEIT(K)=AXWT(IAX,ITY)*WGT(I)
187               XPOS(K)=POSOLD - AXPOS(IAX,ITY) - BRST
188               DXPOS(K)=POS(I) - POSOLD
189               ACCLR(K)=ACC(I)
190               K=K+1
191           110 CONTINUE
192           120 CONTINUE
193               IF(IFD.EQ.IBAK(1)) GO TO 11
194               IFD=IFWD(IFD)
```

```
195               GO TO 5
                  C
196                11 IF(ND.EQ.1) GO TO 21
                  C REVERSE DIRECTION
                  C
197               JFD=JFWD(1)
198                15 J=JNDX(JFD)
199               MU = J
200               POSOLD=POS(J)
201               IF (ABS(ACC(MU)).GT.ACCEL)  ACC(MU) = SIGN(ACCEL,ACC(MU))
202               POS(J)=POS(J)+SPD(J,1)*DELTIM+HAFDEL*ACC(J)*DELTIM
203               SPD(MU,1)=ACC(MU)*DELTIM + SPD(MU,1)
204               IF(ABS(SPD(MU,1)).GT.SPDMAX) SPD(MU,1)=SIGN(SPDMAX,SPD(MU,1))      01
                  C
205               IF(ITYPE(J).EQ.1) GO TO 199
206               IF(POS(J).LT.BREND.AND.POSOLD.GT.BRST)  NEWTRK = .TRUE.
207               IF(POSOLD.LT.BREND   .OR.POS(J).GT.BREND)     GO TO 19
                  C
208               IF(TRUCK) GO TO 17
209               TRUCK=.TRUE.
210               IEVENT = IEVENT + 1
211               NTRUK = 1
                  C
212               IF (MPLTON.GT.10)  MPLTON = 10
213               IDPLTN(MPLTON) = IDPLTN(MPLTON) + 1
214               MPLTON = 1
215               GO TO 18
                  C
216                199 CONTINUE
217               IF(POSOLD.LT.BREND   .OR.POS(J).GT.BREND)     GO TO 19
218               DISTTY(1) = DISTTY(1) + 1
219               GO TO 19
                  C
220                17 CONTINUE
221               MPLTON = MPLTON + 1
222               NTRUK = NTRUK + 1
                  C
                  C  DISTRIBUTE TYPE AND LOADS
                  C
223                18 CONTINUE
224               ITY=ITYPE(J)
225               KTYPE(NTRUK) = ITY
226               WGTT(NTRUK) = WGT(J)
227               SPDT(NTRUK) = SPD(1,J)
228               KLANE(NTRUK) = LANE(J)
229               TIMET(NTRUK) = TOTIM
230               DISTTY(ITY)=DISTTY(ITY)+1
231               ILD=WGT(I)/DELWGT
232               IF (ILD.GT.LV) ILD = LV
233               DISTLD(ILD)=DISTLD(ILD)+1
234                19 CONTINUE
                  C
                  C  COLLECT BRIDGE LOADING DATA
                  C
```

```
235             IF(POS(J).GT.BREND)    GO TO 142
        C
236             ITY=ITYPE(J)
237             NAX=NOAX(ITY)
238             POSS = POSOLD + VEHLEN(ITY)
239             IF(POSS.LT.BRST)  GO TO 142
        C
240             DO 140 IAX=1,NAX
241             LNUM(K) = -LANE(J)
242             WEIT(K)=AXWT(IAX,ITY)*WGT(J)
243             XPOS(K)=POSOLD + AXPOS(IAX,ITY) - BRST
244             DXPOS(K) = POS(J) - POSOLD
245             ACCLR(K)=ACC(J)
246             K=K+1
247           140 CONTINUE
        C
248           142 CONTINUE
249               IF(JFD.EQ.JBAK(1)) GO TO 21
250               JFD = JFWD(JFD)
251               GO TO 15
252            21 CONTINUE
253               TRUCK=NEWTRK
254               CALL SORPOS
        C
255               IF (DBUG)  CALL GRAPH
        C
256               IF(.NOT.TRUCK) K = 1
257               K = K - 1
258               NOAXL = K
259               IF (K.GT.0)  DATABL = .TRUE.
260               IF (K.EQ.0)  GO TO 150
261               IF(.NOT.DBUG)  GO TO 150
262               WRITE(6,1145) IEVENT, TOTIM
263               DO 146 I=1,K
264           146 WRITE(6,1146) I,LNUM(I),WEIT(I),XPOS(I),DXPOS(I),ACCLR(I)
        C
265               WRITE(6,1147) NTRUK
266          1147 FORMAT ('0TRUCKS THIS EVENT ',I4,//' TRUCK  TYPE    WEIGHT   ',
                    1    'SPEED  LANE  TIME ON BRIDGE')
267               DO 147 I=1,NTRUK
268           147 WRITE (6,1148) I,KTYPE(I), WGTT(I), SPDT(I), KLANE(I), TIMET(I)
269          1148 FORMAT(1X,I5,I6,F10.0,F8.0,I6,F8.1)
        C
270          1145 FORMAT('0EVENT NO.',I6,
                    1      '   BRIDGE LOAD TIME =',F12.4,'SEC'//10X,'LANE',4X,'WEIGHT  '
                    1,'POSITION',2X,'DISTANCE',2X,'ACCELERATION'/)
271          1146 FORMAT(7X,I2,3X,I2,F10.0,F10.2,F10.2,F14.2)
272           148 CONTINUE
273           150 RETURN
274               END
```

REFERENCES

1. "Forecasting of Heavy Loading Patterns on Highway Bridges," H. Bissell, et al, Federal Clearing House No. PB193119, Kelly Scientific Corp.

GPO 873-6